Normandie Université

UNIVERSITÉ LE HAVRE NORMANDIE

THÈSE

Pour obtenir le diplôme de doctorat

Spécialité **INFORMATIQUE**

Préparée au sein de l'**Université Le Havre Normandie**

Adaptation dynamique de la gestion de charge computationnelle par criticalité auto-organisée

Présentée et soutenue par

HELEINE PAULIN

Thèse soutenue le 19/09/2025

devant le jury composé de :

M. OLIVIER DAMIEN	Professeur des Universités - ULHN - Université Le Havre Normandie	Directeur de thèse
M. LANG CHRISTOPHE	Professeur des Universités - Université Bourgogne Franche-Comté	Président du jury
M. JIMENEZ LAREDO JUAN LUIS	Professeur (dans un établissement à l'étranger) - UNIVERSIDAD DE GRANADA	Co-encadrant
M. MARILLEAU NICOLAS	Ing. Rech-HDR. IRD Bondy - UMMISCO Unité de Modélisation Mathématique et Informatique des Systèmes Complexes	Membre du jury
MME JOHNEN COLETTE	Professeur des Universités - Université de Bordeaux	Rapporteur du jury

Thèse dirigée par **OLIVIER DAMIEN** (LABORATOIRE D'INFORMATIQUE DE TRAITEMENT DE L'INFORMATION ET DES SYSTEMES)

MI∫IΣ

litis

Remerciements

Je tiens à exprimer ma profonde gratitude envers toutes les personnes qui, de près ou de loin, ont contribué à la réalisation de ce travail de thèse.

Je souhaite tout d'abord remercier chaleureusement Damien OLIVIER et Juan Luis JIMÉNEZ LAREDO, respectivement directeur et co-encadrant de ma thèse. Damien, tu as su m'orienter vers la voie académique à l'issue de ma licence d'informatique, alors que je m'interrogeais encore sur mon avenir. Cette orientation a conduit à un stage de recherche, puis à la présente thèse. Malgré les événements inattendus, parfois regrettables, qui ont jalonné ce parcours, vous avez tous deux maintenu un encadrement attentif et de grande qualité. Je vous en suis sincèrement reconnaissant et je suis persuadé que vous faites partie des meilleurs encadrants que l'on puisse avoir.

Je tiens également à exprimer ma gratitude aux membres du jury, et tout particulièrement aux rapporteurs, Colette JOHNEN et Christophe LANG, pour avoir accepté de lire attentivement ce manuscrit et de l'évaluer. Vos rapports, d'une grande bienveillance, m'ont profondément touché. J'adresse aussi mes remerciements à Nicolas MARILLEAU, qui a accepté de se joindre à cette aventure en tant que membre du jury.

Le retour de Juan Luis dans sa patrie m'a offert l'opportunité de séjourner à Grenade, en Espagne, pendant un mois à ses côtés. Ce voyage m'a apporté énormément, tant sur le plan scientifique que personnel. Je lui en suis de nouveau reconnaissant, et j'adresse également mes remerciements à l'Université de Grenade pour m'avoir accueilli dans les locaux du CITIC durant ce séjour. J'exprime aussi toute ma gratitude à Éric SANLAVILLE qui, en tant que co-directeur du LITIS, a assuré le financement de ce voyage.

Je souhaite ensuite remercier toute l'équipe du LITIS du Havre, que j'ai d'abord côtoyée en tant qu'étudiant au cours de mon parcours universitaire, avant de la rejoindre comme collègue. L'ambiance bienveillante qui y règne, faite de sérieux mais aussi de nombreux éclats de rire, a rendu ces années de travail particulièrement agréables.

J'adresse une pensée particulière à Vincent, avec qui j'ai partagé mon bureau pendant quatre années, de nombreuses discussions et même quelques échanges de coups… à la boxe, en toute bienveillance bien sûr ! Je n'oublie pas non plus les membres de feu la B101 (aujourd'hui B105), avec qui les parties de tarot enflammées et les jeux de plateau du midi ont rythmé les pauses.

Enfin, je tiens à remercier du fond du cœur mes parents, dont le soutien indéfectible m'a accompagné tout au long de ces quatre années de thèse.

Avertissement

Dans le cadre de la rédaction de ce manuscrit, plusieurs outils ont été utilisés dans le but d'en améliorer la qualité et de faciliter le processus de travail.

Tout d'abord, la plateforme Overleaf a été utilisée pour la rédaction du document. Overleaf est un éditeur LaTeX en ligne qui facilite grandement la collaboration, en particulier avec les encadrants.

Ensuite, des outils d'intelligence artificielle basés sur des modèles de langage (tels que ChatGPT, Le Chat ou NotebookLM) ont été employés pour des tâches de relecture, de reformulation linguistique et d'amélioration stylistique. Leur utilisation a permis de renforcer la clarté et la fluidité de certaines sections du texte. Ils ont également été sollicités pour la synthèse d'articles scientifiques, dans le but d'optimiser la phase de revue de la littérature.

Il est important de souligner que ces outils ont été utilisés exclusivement comme aide à la rédaction et à la productivité. Aucun contenu scientifique original n'a été généré par ces systèmes. Toutes les idées, interprétations et conclusions présentées dans ce document sont le fruit d'un travail personnel. Le contenu de la revue de la littérature a été étudié, compris, puis rédigé avant l'intégration dans le document.

Enfin, dans un souci de transparence et de reproductibilité, le code source utilisé pour obtenir les résultats présentés dans ce manuscrit est mis à disposition en open source sous licence MIT. Il est librement accessible afin que chacun puisse le consulter, l'utiliser ou le modifier, contribuant ainsi à la vérifiabilité des résultats et à la poursuite des travaux par la communauté scientifique.

Résumé

De nombreux systèmes, naturels ou artificiels, s'appuient sur des mécanismes d'équilibrage de charge pour fonctionner efficacement, mécanismes qui dépendent directement de l'organisation de leurs composants. Cette organisation peut être centralisée, contrôlée par une entité unique, ou émerger à partir de décisions prises localement, conduisant à une auto-organisation du système. Nous nous intéressons dans cette thèse à la criticalité auto-organisée, un phénomène où des instabilités locales génèrent spontanément des organisations. Nous explorons ainsi comment ce phénomène peut être exploité pour équilibrer la charge dans des systèmes informatiques distribués. Dans un premier temps, nous examinons la robustesse des systèmes qui présentent de la criticalité auto-organisée à l'aide du modèle du tas de sable proposé par Bak, Tang et Wiesenfeld. Nos résultats montrent que l'introduction d'une quantité minimale d'aléatoire dans la structure du système augmente notablement sa résistance aux défaillances, repoussant ainsi les seuils critiques d'effondrement. Dans un second temps des mécanismes d'auto-adaptation, utilisant un modèle dérivé du tas de sable où chaque élément est susceptible de traiter des tâches, sont développés. Ces mécanismes s'adaptent efficacement aux surcharges tout en présentant des atouts et des limites propres. Ces travaux ouvrent des perspectives vers des systèmes distribués robustes et adaptatifs inspirés de l'auto-organisation.

Mots-clé : Système complexe, criticalité auto-organisée, auto-organisation, tas de sable, équilibrage de charge, robustesse, auto-adaptation.

Abstract

Many natural and artificial systems rely on load-balancing mechanisms to operate efficiently, mechanisms that are directly influenced by the organization of their components. This organization can be either centralized, governed by a single controlling entity, or it can emerge from local decision-making processes, leading to self-organization within the system. This dissertation focuses on self-organized criticality, a phenomenon in which local instabilities spontaneously give rise to organized behavior. We investigate how this phenomenon can be leveraged to balance load in distributed computing systems. First, we examine the robustness of systems exhibiting self-organized criticality through the sandpile model introduced by Bak, Tang, and Wiesenfeld. Our results show that introducing a minimal amount of randomness into the system's structure significantly enhances its resilience to failures, thereby increasing the critical thresholds at which collapse occurs. Second, we develop self-adaptive mechanisms based on a modified sandpile model, in which each component is capable of processing tasks. These mechanisms adapt efficiently to overload conditions while exhibiting specific advantages and limitations. This work opens new perspectives for the design of robust and adaptive distributed systems inspired by self-organization.

Key-words : Complex system, self-organized criticality, self-organization, sandpile, load balancing, robustness, self-adaption.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

L'étude des phénomènes naturels a depuis longtemps inspiré le développement de modèles computationnels capables de résoudre des problèmes complexes. Qu'ils soient biologiques, physiques ou comportementaux, ces phénomènes présentent des dynamiques robustes, adaptatives et souvent décentralisées. Cette observation a conduit à l'émergence de l'informatique bio-inspirée : une discipline interdisciplinaire qui exploite les principes évolutifs, collectifs et auto-organisés observés dans la nature pour concevoir des algorithmes.

Ces systèmes naturels se distinguent par leur capacité d'auto-organisation, où un comportement global émerge de simples interactions locales. On peut citer les mouvements synchronisés d'oiseaux et de poissons (REYNOLDS, 1987), l'organisation des colonies de fourmis et d'abeilles pour la recherche de nourriture (BONABEAU et al., 1997), ainsi que la criticalité auto-organisée illustrée par les avalanches dans les milieux granulaires (BAK et al., 1987) et les séismes (STEIN & KLOSKO, 2002).

Cette capacité à atteindre un équilibre dynamique sans contrôle centralisé, grâce aux interactions locales et aux phénomènes critiques, ouvre la voie à l'application de ces principes aux infrastructures numériques. Tout comme les avalanches de sable redistribuent de l'énergie ou que les comportements collectifs optimisent la recherche de ressources, on peut envisager des protocoles décentralisés où la charge de travail se régule par des ajustements locaux.

Dans le contexte numérique actuel, la fourniture de services informatiques fiables, performants et durables représente un défi majeur. La croissance exponentielle de la demande rend indispensable le développement de mécanismes d'équilibrage de charge efficaces et décentralisés, capables de s'adapter à des environnements dynamiques tels que les datacenters, les infrastructures cloud ou les réseaux de capteurs. Le modèle du tas de sable, introduit par Bak, Tang et Wiesenfeld (BAK et al., 1987) est un exemple paradigmatique de criticalité auto-organisée appliquée à un système discret simple et ouvert : il montre comment, par l'action locale de cellules souples à un seuil critique, le système atteint spontanément un état de tension critique ponctué d'avalanches qui le réorganisent. Récemment, plusieurs travaux ont exploré l'application du modèle du tas de sable pour répartir la charge de manière locale et émergente (GĄSIOR & SEREDYŃSKI, 2017 ; LAREDO et al., 2017 ; LAREDO et al., 2014). Toutefois, ces études se limitent généralement à des grilles régulières et négligent à la fois la diversité des topologies et la résilience aux défaillances, tout en se basant sur des hypothèses fortes

comme le nombre illimité de ressources.

Cette thèse adopte une approche centrée sur l'analyse et l'extension de modèles bio-inspirés pour l'équilibrage de charge décentralisé, en s'appuyant sur le principe de criticalité auto-organisée. Le mécanisme d'avalanches critiques, hérité du modèle du tas de sable, est mobilisé pour définir des protocoles locaux où l'injection ou la redistribution de charge déclenche spontanément des réajustements émergents globaux, assurant ainsi une adaptation continue et robuste face aux variations de la demande.

Ces différentes observations ont orienté le travail de recherche qui se structure autour de deux contributions principales :

— **Première contribution :** une étude approfondie de la robustesse du modèle du tas de sable sur des structures variées. Un cadre expérimental original intègre un algorithme de recâblage pour générer des graphes petit-monde et un mécanisme de suppression aléatoire de cellules, afin d'évaluer l'influence de la topologie et la robustesse aux pannes sur la criticalité du système.

— **Deuxième contribution :** prolongement d'une extension du modèle du tas de sable qui introduit des grains avec une taille et des cellules ayant une capacité de traitement quelque soit la situation sur la grille, simulant ainsi des ressources réparties traitant progressivement les tâches. Plusieurs mécanismes décentralisés d'ajustement dynamique de la capacité de traitement sont proposés et comparés, dans le but d'optimiser un compromis entre consommation énergétique et qualité de service dans différents scénarios avec un nombre borné de ressources.

Au-delà de ces apports, la thèse propose un cadre méthodologique et illustre comment les principes de criticalité auto-organisée peuvent être exploités pour concevoir des systèmes distribués à la fois adaptatifs, robustes et efficients.

1.1 Structure du document

Cette thèse est structurée en plusieurs chapitres répartis en deux grandes catégories : les chapitres documentaires, qui dressent un état de l'art sur les thématiques étudiées, et les chapitres de contributions, qui présentent les travaux de recherche menés.

Le **chapitre introductif**, correspondant au présent chapitre, propose une entrée en matière aux thématiques abordées dans ce mémoire. Il fournit également des informations pratiques telles que la structure globale du document et l'accessibilité du code source développé pour les simulations.

Le **deuxième chapitre** constitue un état de l'art des mécanismes d'équilibrage de charge. Il explore les paradigmes qui caractérisent les systèmes d'équilibrage, notamment la nature de l'environnement dans lequel ils opèrent, leur architecture de contrôle, ainsi que leur mode de prise de décision. Ce chapitre introduit également un ensemble de métriques permettant d'évaluer les performances de ces mécanismes. Une taxonomie des algorithmes d'équilibrage est proposée, accompagnée d'exemples représentatifs pour chaque classe. L'attention se porte ensuite plus spécifiquement sur les mécanismes d'auto-organisation, qui apparaissent particulièrement pertinents dans le cadre de l'équilibrage de charge décentralisé.

Le **troisième chapitre** est consacré au concept de criticalité auto-organisée, largement observé dans les systèmes naturels. Pour l'étudier, nous nous intéressons aux modèles qui permettent de la reproduire, en particulier le modèle du tas de sable de Bak, Tang et Wiesenfeld, qui constitue la base de nos travaux. Ce second chapitre documentaire explore également l'influence de la topologie sur la criticalité auto-organisée et la robustesse qu'elle confère aux systèmes. Enfin, il examine son application à l'équilibrage de charge dynamique à travers le prisme du modèle du tas de sable.

Le **quatrième chapitre** présente notre première contribution : une étude de la robustesse du modèle du tas de sable. Nous y introduisons un cadre d'expérimentation original permettant d'observer l'évolution, à la fois structurelle et dynamique, du modèle dans des environnements variés (grille régulière, graphe petit-monde, structure aléatoire) et dégradés. L'introduction de perturbations dans la structure initiale se révèle capable de repousser significativement le seuil d'effondrement du système.

Le **cinquième chapitre**, correspondant à notre seconde contribution, introduit une contrainte sur l'environnement d'exécution du modèle du tamis, une extension du tas de sable adaptée à la modélisation de systèmes de traitement de tâches. Après avoir posé les problématiques et les leviers d'action envisageables, nous présentons plusieurs approches permettant de rendre le modèle auto-adaptatif face aux surcharges induites par cette limitation. Ces approches sont d'abord étudiées individuellement, puis comparées dans différents scénarios, à la fois artificiels et réalistes. Toutes sont fonctionnelles, chacune présentant des avantages et des limites.

Enfin, le **dernier chapitre** propose une conclusion générale. Il récapitule et discute les principaux apports du travail effectué, et ouvre des perspectives pour de futures recherches.

Une annexe est également fournie, regroupant sous forme de tableaux les différentes simulations réalisées et les résultats obtenus.

1.2 Accessibilité du code des simulations

Le code développé dans le cadre de cette thèse pour produire les résultats présentés est mis à disposition en open source sous licence MIT sur le dépôt Git suivant : https://git.litislab.fr/pheleine/self-adaptive-sand-sieve. Les données externes libres utilisées lors des expérimentations, notamment les traces d'exécution issues de certains systèmes du projet libre d'accès Grid Workload Archive (IOSUP et al., 2008a) (https://www.atlarge-research.com/gwa.html), sont également incluses dans le dépôt. La structure du code, ainsi que les modalités d'utilisation du code et des données, sont détaillées en Annexe B.

Chapitre 2

Équilibrage de charge

Table des matières du chapitre

L'**équilibrage de charge** est une stratégie visant à répartir la charge de travail au sein d'un système afin d'en utiliser au mieux ses **ressources limitées**. Il existe de nombreuses situations nécessitant de l'équilibrage de charge autour de nous. Prenons l'exemple de l'encaissement dans un supermarché. Les clients (charge de travail) vont se répartir sur les différentes caisses (ressources) pour payer leurs futurs achats. Le personnel de caisse est d'une efficacité variable pour encaisser les articles, tandis que les clients ont un panier plus ou moins rempli. Ces deux caractéristiques correspondent respectivement à la capacité des ressources et à la quantité de charge entrante dans le système. Les clients vont naturellement se diriger vers les caisses les moins chargées, puis, si leur file d'attente progresse lentement, changeront de caisse pour une plus rapide dans l'espoir que leur tour vienne plus tôt. En outre, les caisses peuvent ouvrir ou fermer selon le nombre de clients. Nous avons ici l'exemple d'un système complexe dont l'environnement est dynamique (nombre de caisses, vitesse d'encaissement, nombre de clients et taille des paniers) et dont la charge s'auto-équilibre (arrivée des clients sur les caisses les moins chargées et changement de caisse). Par cet exemple, nous touchons du doigt la richesse, mais également la complexité, du monde de l'équilibrage de charge. Le choix d'un algorithme d'équilibrage n'est donc pas anodin et est, le plus souvent, multi-critères par rapport aux spécificités du système.

Dans ce chapitre, nous commencerons par aborder les paradigmes des systèmes d'équilibrage en Section 2.1, afin de présenter les concepts fondamentaux des algorithmes sous-jacents. Chaque système dispose de ses propres spécificités et contraintes. Nous continuerons donc en Section 2.2 avec les métriques utilisées pour mesurer les performances des différentes solutions d'équilibrage. Puis, nous nous attarderons sur une classification des stratégies de répartition d'un point de vue algorithmique en Section 2.3. Nous nous focaliserons ensuite, en Section 2.4, sur l'auto-organisation et son utilisation pour équilibrer la charge dynamiquement. Pour terminer, en Section 2.5, nous discuterons des points forts et des points faibles des méthodes présentées.

2.1 Paradigmes des systèmes d'équilibrage

Les caractéristiques d'un système déterminent le choix de la stratégie de répartition de charge pouvant être mise en œuvre. Par ailleurs, les orientations conceptuelles des systèmes permettent de classifier les algorithmes selon différents paradigmes, bien que ces derniers coexistent simultanément. Nous commencerons par différencier la nature de l'environnement du système : statique ou dynamique. Puis, nous nous intéresserons aux différentes architectures de contrôle de la charge : centralisée, semi-centralisée et décentralisée. Nous terminerons en abordant le mode de prise de décision de l'algorithme.

2.1.1 Nature de l'environnement

Le paradigme de la nature de l'environnement concerne la manière dont les algorithmes sont conçus en fonction des caractéristiques du système dans lequel ils opèrent. L'environnement d'exécution impacte fortement le choix de la stratégie d'équilibrage puisque les contraintes et les objectifs sont directement corrélés à la stabilité et l'évolutivité du système. Il existe deux catégories d'environnements (ALAKEEL, 2009 ; DEEPA & CHEELU, 2017) : les statiques et les dynamiques. La catégorisation est effectuée en fonction des variations de la charge entrante et des ressources. Nous étudierons en premier les caractéristiques des environnements statiques, puis celles des environnements dynamiques. Nous verrons également, pour finir, un dérivé d'environnement dynamique à mi-chemin entre les deux. Le Tableau 2.1 résume les caractéristiques de ces trois catégories.

Environnement	**Ressources (nombre et capacités)**	**Charge entrante**
Statique	Fixes	Fixe ou connue à l'avance
Dynamique	Fluctuantes	Fluctuante
Hybride	Initialement fixes, fluctuantes au besoin	Attendue, mais peut fluctuer

TABLE 2.1 – Résumé des caractéristiques des environnements.

2.1.1.1 Environnement statique

Dans un environnement statique, les ressources et la charge à équilibrer sont fixes. Il n'y a pas ou peu de variations dans le temps. Les décisions d'équilibrage peuvent donc être adaptées en amont directement au nombre de ressources disponibles, à leur capacité, ainsi qu'à la charge attendue. La Figure 2.1 présente une schématisation de l'environnement statique.

Les stratégies d'équilibrage pour ce type d'environnement sont très souvent déterministes et de faible complexité. Les algorithmes varient cependant dans leurs approches. On trouve de la répartition approximative avec des méthodes naïves (distribution égale ou équitable par exemple), des heuristiques (règles empiriques pour guider la répartition) ou encore l'optimalité (optimisation complète pré-fonctionnement). Nous nous pencherons en détails sur ces approches en Section 2.3.1.

Un bon exemple d'environnement statique est une usine. Toute la chaîne de production est préparée et optimisée en amont durant la conception du système (optimalité). Chaque poste de travail au sein de l'usine

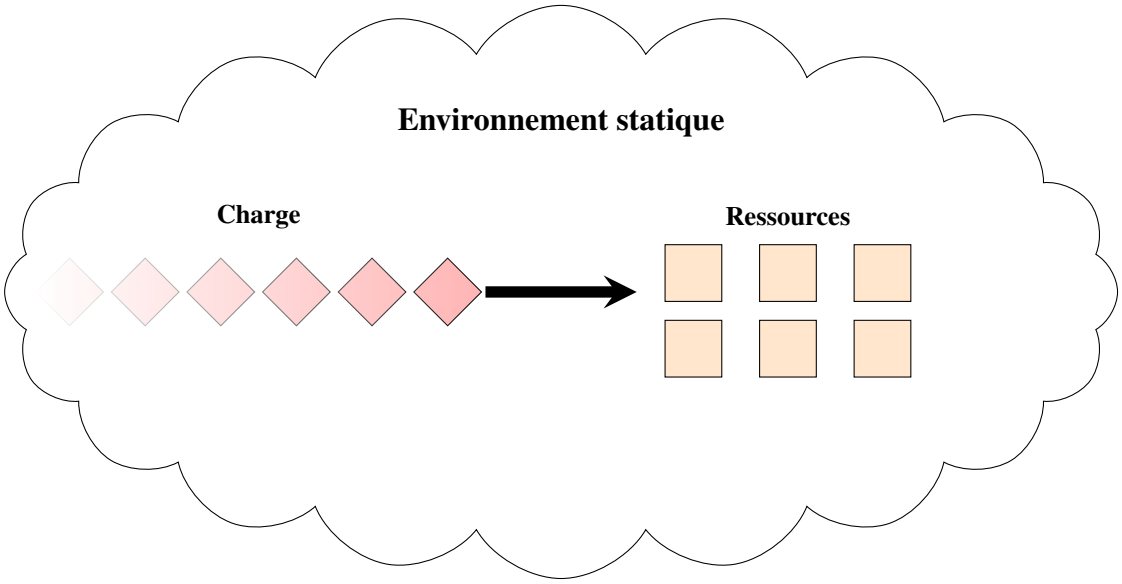


FIGURE 2.1 – Schématisation d'un environnement statique : la charge entrante, le nombre de ressources et leurs capacités sont fixes.

exécute une tâche spécifique à une cadence prédéfinie afin qu'aucun temps mort ne survienne. Tout le système est figé et un changement dans celui-ci, tel que l'ajout d'un poste de travail, nécessite de recalculer la solution optimale.

2.1.1.2 Environnement dynamique

Dans un environnement dynamique, que ce soit le nombre de ressources, leurs capacités, ou bien la charge, tout peut fluctuer. L'évolutivité de l'environnement rend l'équilibrage bien plus complexe puisqu'il nécessite une adaptation continue au changement de situation. Pour simplifier, à l'inverse d'un environnement statique, les algorithmes doivent fonctionner efficacement peut importe la charge de travail et les ressources. La Figure 2.2 présente une schématisation de l'environnement dynamique.

Le problème de l'équilibrage de charge se complexifie à cause de la nature évolutive de l'environnement. Les stratégies sont donc plus nombreuses et varient d'autant plus dans leurs approches. Les algorithmes se divisent généralement en trois catégories d'après leur architecture de contrôle : algorithmes centralisés, semi-centralisés et décentralisés. Nous nous intéresserons à ce paradigme en Section 2.1.2, tandis que nous étudierons différents algorithmes selon leur type de stratégie en Section 2.3.2.

L'exemple d'environnement dynamique le plus pertinent de nos jours est certainement les centres de données (HE et al., 2015 ; ZHANG et al., 2017), souvent qualifiés par leur nom anglais *data centers*. La charge, inconnue et fluctuante, doit être efficacement distribuée sur les serveurs d'un centre. La distribution des tâches doit être optimisée pour augmenter la longévité du matériel en évitant un fonctionnement à plein régime en permanence, pour réduire la consommation énergétique ou encore pour que le système ait un temps réponse faible. Si un serveur tombe en panne, il ne doit plus être compté dans la répartition de charge et ses tâches doivent être redistribuées. À l'inverse, lorsqu'un serveur est ajouté, il doit automatiquement être inclus dans

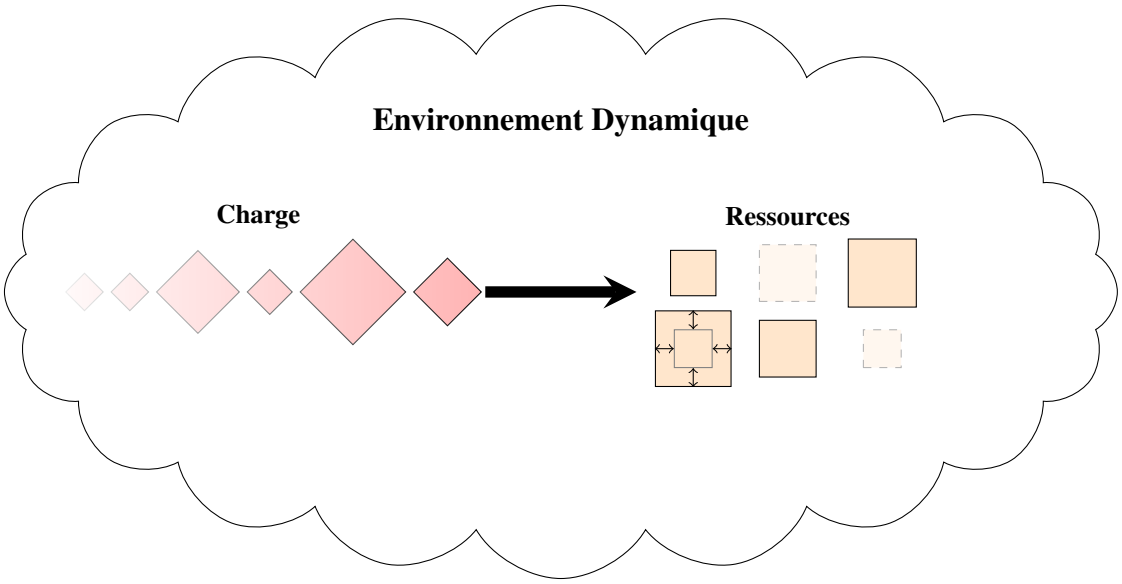


FIGURE 2.2 – Schématisation d'un environnement dynamique : la charge entrante, le nombre de ressources et leurs capacités fluctuent.

l'écosystème du centre.

2.1.1.3 Environnement hybride

Un environnement hybride est un dérivé de système dynamique à mi-chemin entre statique et dynamique. La charge de travail est attendue, bien qu'elle puisse évoluer, et les ressources sont préparées en conséquence. Lorsqu'une surcharge inattendue survient, de nouvelles ressources sont automatiquement allouées pour y faire face. À contrario, lorsque la charge diminue, les ressources excédentaires sont automatiquement désallouées pour retrouver la configuration initiale. Pour résumer, un nombre initial de ressources est alloué par rapport à la charge attendue et leur nombre (ou capacité) est ajusté au besoin.

Le système de caisses d'un supermarché, utilisé comme exemple en introduction du chapitre, illustre bien ce type d'environnement. Un nombre minimal de caisses est constamment ouvert afin de garantir un service de base, ce nombre étant défini selon l'affluence attendue à différents moments de la journée. En cas de forte affluence, du personnel en réserve est mobilisé pour ouvrir de nouvelles caisses. Les clients se redistribuent alors spontanément entre les caisses disponibles, ce qui permet de fluidifier le traitement des files. Une fois la vague passée, les caisses superflues ferment et le personnel retourne en réserve.

Ce type d'environnement est particulièrement représentatif des systèmes cloud, et plus spécifiquement des services web. L'usage de ces services est en général prévisible, mais le système doit pouvoir s'adapter à des variations soudaines de trafic. Dans ce cadre, des technologies d'orchestration comme Kubernetes ont été développées. Elles offrent notamment des fonctionnalités de mise à l'échelle automatique, permettant d'ajuster dynamiquement le nombre de répliques d'un service en fonction de la charge observée (THE KUBERNETES AUTHORS, 2025). Les réactions du système sont prédéfinies mais paramétrables dynamiquement, offrant ainsi un compromis efficace entre flexibilité et contrôle.

2.1.2 Architecture de contrôle

Le paradigme de l'architecture de contrôle définit qui prend les décisions de distribution de la charge dans le système. Les architectures sont au nombre de trois (AL-RAYIS & KURDI, 2013 ; IVANISENKO & RADIVILOVA, 2015) : centralisée, semi-centralisée et décentralisée. Chacune dispose de forces et de faiblesses et conviennent à des situations différentes. C'est ce que nous allons étudier dans cette section, dans l'ordre énoncé précédemment.

2.1.2.1 Centralisation

Le principe de l'architecture centralisée est d'avoir un composant unique de contrôle qui prend les décisions pour tout le système. Ce contrôleur central dispose d'une vue centrale et interagit avec toutes les ressources disponibles. La structure sous-jacente est typiquement une étoile dont le contrôleur en est le centre et les ressources ses extrémités. La duplication de bases de données ou le calcul parallèle utilisent cette méthode désignée par le pattern maître-esclave. Dans le premier cas, le serveur central (maître) va propager les requêtes d'écriture sur toutes les bases de données dupliquées (esclaves). Dans le deuxième cas, le processeur "maître" divise les calculs complexes qu'il distribuera aux processeurs "esclaves" avant d'en agréger les résultats. On notera que dans le cas d'un système centralisé, toutes les ressources doivent être en communication avec le contrôleur central, ce qui n'est pas le cas desdites ressources entre elles. La Figure 2.3 illustre la centralisation avec un contrôleur (maître) qui communique avec trois ressources (esclaves). Dans le cadre de l'équilibrage de charge, tout transite par le contrôleur central qui décide sur quelle ressource envoyer les tâches pour équilibrer au mieux l'utilisation des ressources. Le web utilise cette architecture à plusieurs niveaux pour répartir le trafic entre les services d'un serveur web, mais également entre différentes instances de celui-ci.

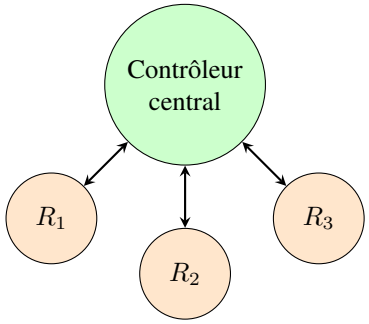


FIGURE 2.3 – Schématisation de la centralisation.

2.1.2.2 Semi-centralisation

La semi-centralisation, aussi qualifiée d'architecture hiérarchique, consiste généralement en de la centralisation mise en arborescence. Chaque nœud de l'arbre correspond à un contrôleur qui répartit la charge envoyée par son parent entre ses fils, tandis que les feuilles correspondent aux ressources. Un contrôleur au niveau n connaît donc l'état de ses ressources (sous-contrôleurs ou ressources directes) au niveau $n+1$ et remonte l'information de son état à son contrôleur parent du niveau $n-1$ s'il existe. Par conséquent, un contrôleur prend les

décisions pour un nombre restreint de sous-contrôleurs, qui eux-mêmes contrôlent une sous-partie du système, et ainsi de suite. Au final, la répartition de la charge s'effectue de concert au travers de tous les niveaux de la hiérarchie. Cette architecture est qualifiée de semi-centralisée puisque la prise de décision de chaque branche et sous-branche de l'arbre est centralisée, mais les décisions sont toutefois indépendantes d'une branche à l'autre.

Le web et les services cloud de manière générale reposent sur cette architecture (AFZAL & KAVITHA, 2019). La Figure 2.4 schématise son utilisation appliquée au web. Le niveau 1 correspond aux services tels qu'Amazon Web Services, Google Cloud Platform ou encore Microsoft Azure, qui permettent de dupliquer dynamiquement tout un serveur web selon le trafic pour en répartir la charge. Puis vient le niveau 2 qui correspond aux instances du serveur web dupliqué. Le serveur web, Nginx pour ne citer que le plus utilisé depuis 2020, fait de la répartition de charge entre ses services du niveau 3, préalablement dupliqués si besoin, pour y distribuer les requêtes entrantes.

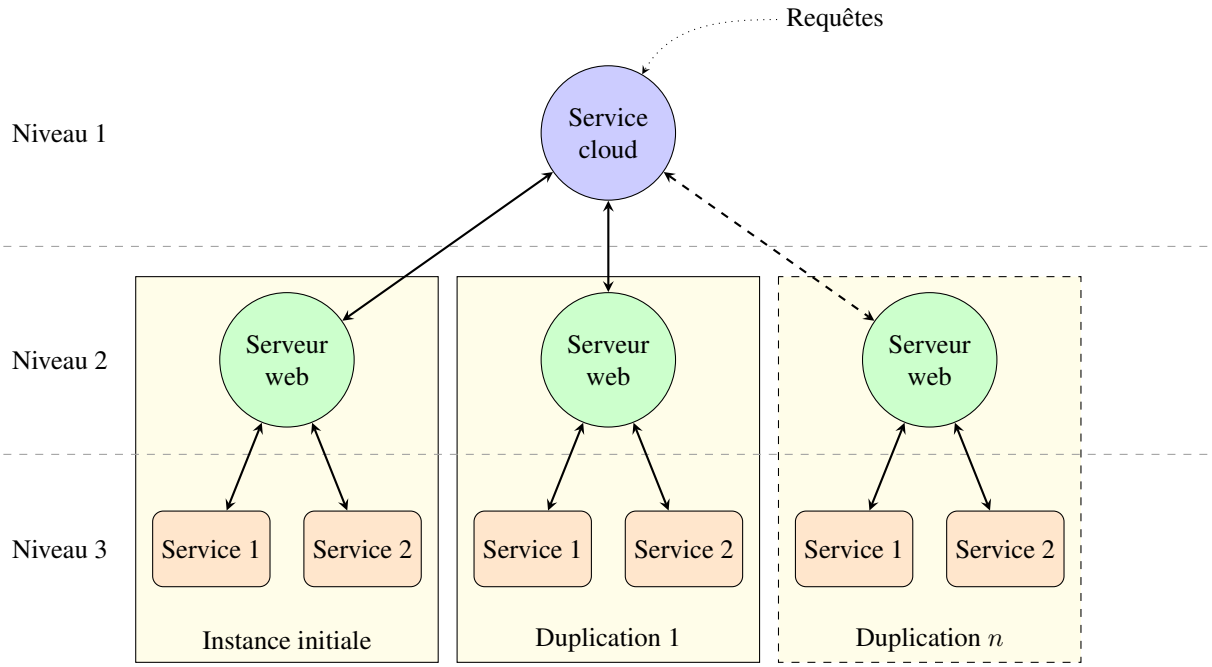


FIGURE 2.4 – Schématisation de l'architecture semi-centralisée (hiérarchique) appliquée au web.

Cette architecture est généralement paramétrée en amont (profondeur de l'arbre, nombre de nœuds contrôleurs, etc.). Cependant, il existe tout de même des algorithmes pour maintenir la structure de l'arbre de manière décentralisée et dynamique tels que les modèles D²-Tree (BRODAL et al., 2015) et D³-Tree (SIOUTAS et al., 2022).

2.1.2.3 Décentralisation

À l'opposé de l'architecture centralisée, l'approche décentralisée se caractérise par l'absence d'un nœud de contrôle unique. Chaque ressource du système fonctionne de manière autonome, prenant ses décisions sur la base de son propre état et, le cas échéant, d'une connaissance partielle ou complète de l'environnement global. Cette organisation permet l'émergence de comportements coopératifs et/ou autonomes, selon les modalités

d'interaction entre les entités.

Cette méthode se base sur l'échange d'information au sein du système par divers algorithmes et protocoles de communication. Par conséquent, les ressources doivent être suffisamment connectées les unes aux autres pour faciliter les interactions. La structure sous-jacente est généralement un graphe fortement connexe dont les distances inter-sommets sont faibles (peu de sommets intermédiaires). Les systèmes peer-to-peer tels que BitTorrent fonctionnent par exemple de cette manière. Les machines du réseau partagent les données déjà téléchargées pour que d'autres les récupèrent et les partagent à leur tour ; chaque machine est à la fois émétrice et réceptrice de l'information.

La Figure 2.5 illustre la décentralisation avec cinq ressources communiquant plus ou moins entre elles. Appliquée à la répartition de charge, cette architecture dite distribuée permet d'équilibrer le travail entre les ressources sans contrôle externe.

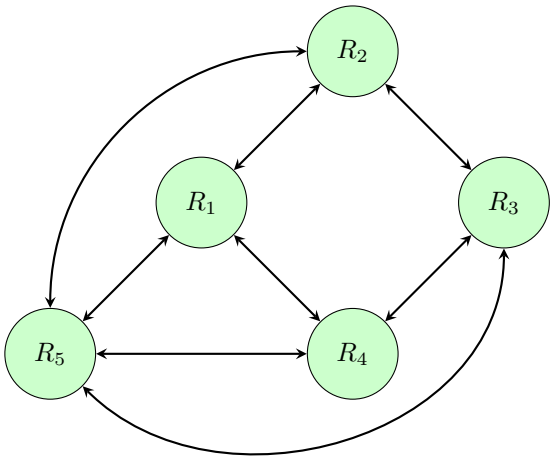


FIGURE 2.5 – Schématisation de la décentralisation.

2.1.3 Mode de prise de décision

Après avoir examiné les différentes architectures de contrôle afin de déterminer l'entité responsable de la prise de décision, il convient désormais d'examiner les mécanismes selon lesquels ces décisions sont élaborées, à travers le prisme des modes décisionnels. Ce paradigme repose sur un ensemble de choix issus de différents critères dont la combinaison détermine la forme finale du processus décisionnel. Nous débuterons cette analyse par la temporalité de la décision, qui peut être prise de manière proactive ou réactive. Nous examinerons ensuite la nature du processus décisionnel, qui peut être déterministe ou stochastique. Nous poursuivrons par l'adaptabilité des algorithmes. Enfin, nous nous intéresserons à la méthodologie de prise de décision, qui repose soit sur des règles préétablies, soit sur des approches d'apprentissage.

2.1.3.1 Temporalité de la décision

Le critère de la temporalité de la décision est crucial dans l'équilibrage de charge, car il définit l'optique dans laquelle une décision est prise : la distribution des tâches peut être soit anticipée soit faite en direct à leur arrivée. On peut qualifier ce critère de réactivité. Les algorithmes se découpent donc en deux approches qui seront détaillées dans cette section : les proactifs et les réactifs.

Algorithmes proactifs Les algorithmes proactifs visent à anticiper les besoins futurs du système en s'appuyant sur des modèles prédictifs issus d'analyses de données historiques. Ces modèles peuvent être fondés sur des méthodes statistiques traditionnelles, telles que les séries temporelles ou les modèles de Markov, ou sur des techniques d'apprentissage automatique, comme les réseaux de neurones ou les machines à vecteurs de support. Par exemple, le modèle proposé dans (YADAV et al., 2021) se base sur de l'apprentissage profond (Long Short-term Memory (HOCHREITER, 1997)) pour prédire la charge future des serveurs à partir de séries temporelles.

Dans certains cas, ces modèles prédictifs sont couplés à des algorithmes d'optimisation, tels que la programmation linéaire ou des algorithmes évolutionnaires, afin d'optimiser la répartition des tâches sur les ressources disponibles. Les auteurs de (BOULMIER et al., 2022) proposent notamment un critère d'équilibrage de charge optimal et automatisé. Celui-ci anticipe le moment optimal pour équilibrer la charge, afin de maximiser la performance de l'algorithme d'équilibrage sous-jacent tout en évitant des opérations inutiles.

Toutefois, tous les algorithmes proactifs ne nécessitent pas une optimisation complexe ; certains reposent sur des heuristiques prédéfinies. Le seuil auto-apprenant proposé par GOLDSZTAJN et al. (2022) et le modèle ULBA (BOULMIER et al., 2019) (Underloading Load Balancing Approach) en sont deux exemples. Le premier ajuste dynamiquement un seuil de répartition de charge (qui anticipe le choix des ressources) selon la charge actuelle des ressources, tandis que le deuxième redistribue la charge des ressources presque surchargées (sélectionnées d'après un score heuristique) pour anticiper leur surcharge.

Enfin, ces systèmes prédictifs sont souvent surveillés en continu via des métriques de performance, et peuvent intégrer des mécanismes d'apprentissage en temps réel pour ajuster leurs décisions en fonction des évolutions du système. C'est notamment le cas du seuil auto-apprenant, du modèle ULBA, ou encore les méthodes à base d'apprentissage par renforcement comme le modèle introduit par MUTHUSAMY et DHANARAJ (2023), basé sur du Q-learning (WATKINS & DAYAN, 1992).

Algorithmes réactifs Les algorithmes réactifs ajustent la répartition des charges en fonction des événements observés dans le système, tels que l'arrivée de nouvelles tâches, la fin de traitement d'une ressource ou une défaillance matérielle. Contrairement aux algorithmes proactifs, ils ne cherchent pas à anticiper les besoins futurs, mais réagissent en temps réel en fonction de l'état du système. Leur simplicité d'implémentation les rend particulièrement adaptés aux environnements où la charge évolue de manière imprévisible.

Certains algorithmes réactifs prennent explicitement en compte la charge du système : par exemple, l'algorithme de moindre connexion (Least-Connection) (MUSTAFA, 2017) assigne systématiquement les nouvelles tâches à la ressource la moins sollicitée. C'est également le cas des stratégies à seuils, pour lesquelles une

action résulte du dépassement d'un seuil de charge, telle que la réaffectation des tâches d'une ressource, ou encore le changement de la ressource cible comme pour la distribution de requêtes tenant compte de la localité (Locality-Aware Request Distribution) (PAI et al., 1998) ou la politique de déchargement distribuée proposée par QIN et al. (2021, 2023a, 2023b). À l'inverse, d'autres approches appliquent des règles de répartition sans considération de l'état du système. C'est notamment le cas de l'algorithme du tourniquet (Round Robin) (HIDAYAT et al., 2020) et de sa variante pondérée (Weighted Round Robin) (DEVI & UTHARIARAJ, 2016), qui distribuent les tâches de manière cyclique entre les ressources.

D'autres techniques issues des modèles multi-agents, plus élaborées, reposent sur une prise de décision décentralisée. Chaque agent alloue des tâches localement en fonction de son voisinage, ce qui peut conduire à une auto-organisation émergente du système, et donc à son équilibrage. L'algorithme d'échantillonnage aléatoire biaisé (Biased Random Sampling) (RAHMEH et al., 2008) illustre bien ce principe : une tâche n'est pas directement affectée à une ressource, mais suit une marche aléatoire contrôlée explorant le voisinage, avant d'être assignée à la ressource dont la charge est la plus faible.

Enfin, bien que les algorithmes évolutionnaires soient souvent utilisés dans les stratégies proactives, certaines adaptations leur permettent de fonctionner en mode réactif, notamment dans des environnements décentralisés et dynamiques. Ces méthodes inspirées de l'intelligence en essaim, comme les colonies de fourmis (Ant Colony Optimization) (LI et al., 2011 ; LIU et al., 2006) ou encore le comportement de recherche de nectar par les abeilles (Honeybee Foraging Behavior) (RANDLES et al., 2010 ; SESUM-CAVIC & KÜHN, 2010a, 2010b), permettent un équilibrage émergent basé sur des heuristiques locales et des interactions entre entités autonomes. Ces méthodes réactives ne planifient pas à l'avance, mais ajustent leur comportement en fonction des variations de charge détectées dans leur environnement immédiat.

2.1.3.2 Nature du processus décisionnel

Une autre manière de qualifier la prise de décision d'un algorithme d'équilibrage de charge réside dans la nature de son processus décisionnel. Le choix entre un modèle déterministe ou stochastique dépend des contraintes du système à équilibrer. Alors que les solutions déterministes offrent stabilité et robustesse dans des environnements bien maîtrisés, les stratégies stochastiques sont souvent privilégiées dans des contextes dynamiques et incertains, où une approche plus flexible et adaptative est nécessaire. Cette section explore ces deux paradigmes, leurs principes, leurs avantages et leurs limites dans le cadre de l'équilibrage de charge.

Processus déterministe Les stratégies déterministes reposent généralement sur des règles fixes, des heuristiques explicites ou des modèles mathématiques prédéfinis pour attribuer une tâche à une ressource donnée. Elles garantissent une prise de décision entièrement prévisible et reproductible : pour un même état du système, la solution d'équilibrage obtenue sera identique. Les critères de sélection des ressources peuvent être soit statiques, soit dynamiques. Dans le premier cas, les choix sont définis à l'avance, indépendamment de l'état courant du système, aboutissant ainsi à une répartition souvent fixe et uniforme des tâches. À l'inverse, dans le second cas, les décisions s'adaptent en fonction d'heuristiques explicites, telles que la charge du processeur ou la latence de connexion. Néanmoins, quelle que soit l'approche adoptée, l'absence d'éléments aléatoires

garantit une réponse systématique et reproductible pour un même ensemble d'entrées. Parmi les algorithmes déterministes statiques, on retrouve notamment le tourniquet (HIDAYAT et al., 2020) et sa variante pondérée (DEVI & UTHARIARAJ, 2016), déjà évoqués dans la section précédente. En ce qui concerne les algorithmes déterministes dynamiques, des stratégies telles que la moindre connexion (MUSTAFA, 2017) ou les approches à seuils (PAI et al., 1998 ; QIN et al., 2021, 2023a, 2023b), également mentionnées auparavant, illustrent cette catégorie. Les algorithmes déterministes constituent une solution simple et efficace pour l'équilibrage de charge dans des environnements maîtrisés, qu'ils soient statiques ou dynamiques mais prévisibles. Cependant, leur rigidité peut s'avérer limitante lorsque la charge varie de manière imprévisible. Afin de mieux s'adapter aux fluctuations dynamiques du système, il existe ainsi des approches intégrant une composante aléatoire, offrant une plus grande flexibilité.

Processus stochastique Contrairement aux algorithmes déterministes, les algorithmes stochastiques intègrent une composante aléatoire dans le processus de prise de décision. Cette approche permet de mieux gérer les systèmes soumis à des charges imprévisibles ou fortement fluctuantes, où une stratégie rigide pourrait conduire à un déséquilibre. L'aléatoire peut être exploité de différentes manières : certains algorithmes sélectionnent une ressource au hasard parmi un sous-ensemble de candidats (comme l'algorithme d'équilibrage aléatoire (AZAR et al., 1994)), tandis que d'autres ajustent dynamiquement leurs choix en fonction de probabilités influencées par l'état du système. Par exemple, l'algorithme d'échantillonnage aléatoire biaisé (RAHMEH et al., 2008) réalise une marche aléatoire sur un graphe représentant les ressources, tout en favorisant celles qui sont sous-utilisées pour l'affectation des tâches. De plus, certaines approches plus avancées utilisent des méthodes bio-inspirées adaptées à l'équilibrage de charge, comme l'optimisation par colonies de fourmis (KATYAL & MISHRA, 2013 ; LI et al., 2011 ; LIU et al., 2006) ou par comportement de recherche de nectar des abeilles (KATYAL & MISHRA, 2013 ; RANDLES et al., 2010 ; SESUM-CAVIC & KÜHN, 2010a, 2010b). Le premier algorithme utilise des agents artificiels qui laissent des traces numériques pour influencer les décisions des agents suivants, tandis que le deuxième biaise l'exploration de l'environnement par un partage des chemins explorés par les agents prédécesseurs pour les recherches futures. Bien que ces méthodes offrent une adaptabilité accrue, elles présentent un coût computationnel potentiellement plus élevé et peuvent nécessiter un ajustement précis des paramètres pour garantir une convergence efficace vers une solution optimale.

2.1.3.3 Adaptabilité

L'adaptabilité désigne la capacité d'un algorithme d'équilibrage de charge à ajuster ses décisions en fonction des variations dynamiques de l'environnement. Un algorithme est considéré comme adaptatif lorsqu'il modifie son comportement en réponse aux fluctuations de charge entrante, aux variations de performance des ressources ou aux événements imprévus tels que des pannes ou des pics de trafic. Cette capacité d'adaptation est souvent liée à la scalabilité, qui désigne la manière dont l'algorithme maintient ses performances lorsqu'il est confronté à une augmentation du nombre de ressources ou de tâches à traiter.

Les méthodes d'adaptation reposent sur différentes approches. Certaines stratégies, comme la moindre connexion (MUSTAFA, 2017) ou les stratégies à seuils (PAI et al., 1998 ; QIN et al., 2021, 2023a, 2023b),

réagissent directement à l'état instantané du système en ajustant la répartition des tâches de manière réactive. D'autres algorithmes exploitent des techniques d'apprentissage automatique, où une rétroaction en temps réel permet d'améliorer progressivement la prise de décision. Enfin, des approches inspirées de la biologie et des systèmes multi-agents, comme l'optimisation par colonies de fourmis (KATYAL & MISHRA, 2013 ; LI et al., 2011 ; LIU et al., 2006), l'optimisation par essaims de particules (Particle Swarm Optimization) (KENNEDY & EBERHART, 1995 ; RAHMEH et al., 2008) ou le comportement de recherche de nectar par les abeilles (KATYAL & MISHRA, 2013 ; RANDLES et al., 2010 ; SESUM-CAVIC & KÜHN, 2010a, 2010b), utilisent des mécanismes d'auto-organisation pour répartir dynamiquement la charge.

L'objectif principal de l'adaptabilité est d'assurer un bon niveau de performance face aux imprévus, ce qui est directement lié à la robustesse de l'algorithme, un concept approfondi en Section 2.2.3. Toutefois, certains algorithmes restent statiques et n'intègrent aucune forme d'adaptation. C'est notamment le cas du tourniquet (HIDAYAT et al., 2020), qui applique une règle fixe en assignant les tâches en séquence, sans prise en compte de l'état du système. Dans des environnements hautement prévisibles et stables, ces algorithmes peuvent s'avérer suffisants, voire préférables, car ils offrent une simplicité de mise en œuvre et un faible coût computationnel, là où l'introduction d'une adaptabilité complexe ne serait pas nécessaire. Cependant, leur manque de flexibilité peut poser problème lorsqu'une scalabilité efficace est requise pour gérer une augmentation de la taille du système. Les stratégies décentralisés offrent notamment de meilleurs résultats dans ce genre de situation.

2.1.3.4 Approche décisionnelle

L'approche décisionnelle d'un algorithme d'équilibrage de charge désigne la manière dont les décisions sont prises et justifiées. Ce critère repose sur la méthodologie employée pour sélectionner la ressource cible lors de la répartition des charges. On distingue principalement deux grandes familles : les approches basées sur des règles explicites et celles qui reposent sur l'apprentissage et l'adaptation dynamique.

À base de règles Les approches basées sur des règles reposent sur des décisions prises à partir de critères prédéfinis, qui peuvent être statiques ou dynamiques, selon qu'elles prennent ou non en compte l'état courant du système. Elles offrent l'avantage d'être simples à implémenter et de nécessiter peu de ressources computationnelles, mais leur rigidité peut les rendre moins efficaces dans des environnements à forte variabilité.

Les règles statiques définissent un comportement immuable, où les décisions sont prises selon des règles fixées à l'avance, indépendamment de l'évolution de la charge ou des ressources disponibles. Elles conviennent particulièrement aux environnements stables et prévisibles, où les variations sont limitées. Un exemple typique est l'algorithme du tourniquet (HIDAYAT et al., 2020), qui distribue les tâches de manière cyclique entre les ressources, garantissant une répartition équitable mais sans tenir compte de la charge réelle. Sa variante, le tourniquet pondéré (DEVI & UTHARIARAJ, 2016), ajuste la distribution en fonction de poids attribués aux ressources, mais sans adaptation en temps réel.

À l'inverse, les règles dynamiques adaptent les décisions en fonction de l'état courant du système, bien qu'elles ne modifient pas leurs propres critères d'évaluation au fil du temps. Elles se basent généralement sur des heuristiques locales, exploitant des métriques telles que la charge des ressources ou leur capacité de

traitement. Par exemple, l'algorithme de la moindre connexion (MUSTAFA, 2017) attribue systématiquement les nouvelles tâches à la ressource la moins sollicitée au moment de la décision. De même, les stratégies à seuils (PAI et al., 1998 ; QIN et al., 2021, 2023a, 2023b) redistribuent la charge lorsque certaines ressources atteignent un niveau d'utilisation critique, permettant un équilibrage réactif. Une autre approche de ce type est l'échantillonnage aléatoire biaisé (RAHMEH et al., 2008), qui utilise un processus aléatoire pour explorer plusieurs ressources voisines avant d'attribuer une tâche. Bien que l'exploration repose sur de l'aléatoire, la décision finale est guidée par une règle heuristique : la tâche est affectée à la ressource rencontrée ayant la plus faible charge. D'autres approches comme First-Fit et Best-Fit appliquent des règles spécifiques : le premier affecte la tâche à la première ressource disponible, tandis que le second sélectionne celle qui offre l'ajustement optimal en fonction de critères définis.

Bien que ces stratégies dynamiques offrent une meilleure réactivité que les méthodes statiques, elles restent limitées par l'absence d'apprentissage : elles s'adaptent instantanément aux changements mais ne réévaluent pas leur propre logique au fil du temps. Cette distinction sera essentielle pour différencier ces approches des méthodes apprenantes, plus flexibles mais également plus coûteuses en ressources computationnelles.

Apprentissage et adaptation dynamique Contrairement aux méthodes reposant sur des règles fixes, les approches basées sur l'apprentissage et l'adaptation dynamique ajustent leurs décisions en fonction des données collectées au fil du temps. Ces méthodes ne se limitent pas à des heuristiques prédéfinies mais modifient progressivement leur comportement en réponse aux évolutions de l'environnement. Elles sont particulièrement adaptées aux systèmes complexes et dynamiques, où la variabilité des charges et des ressources rend inefficace l'usage de règles rigides.

L'apprentissage peut être réalisé selon différentes stratégies. Les modèles d'apprentissage supervisé utilisent des données historiques pour entraîner un modèle capable de prédire la meilleure allocation des tâches (GURES et al., 2022). Les approches par apprentissage par renforcement permettent quant à elles aux algorithmes d'ajuster leurs décisions via un processus d'essais et d'erreurs, en maximisant une récompense définie, comme l'optimisation du temps de réponse ou l'équilibrage des charges entre ressources (MUTHUSAMY & DHANARAJ, 2023). Des techniques comme les réseaux de neurones profonds (Deep Learning) (KAUR et al., 2020 ; YADAV et al., 2021) et les machines à vecteurs de support (Support Vector Machines) (RADHIKA & DURAIPANDIAN, 2021) sont souvent employées pour détecter des tendances complexes dans la charge du système. La mise en place de tels modèles nécessite un entraînement préalable généralement très coûteux en temps et en ressources, mais il peut toutefois être poursuivit en ligne afin de faire évoluer le modèle en fonction du système.

Par ailleurs, des stratégies qui reposent sur des modèles d'intelligence collective et d'auto-organisation, s'inspirent des dynamiques naturelles. Ces approches bio-inspirées adaptées à l'équilibrage de charge décentralisé, comme l'optimisation par essaims de particules (KENNEDY & EBERHART, 1995 ; RAMEZANI et al., 2014), les algorithmes basés sur le comportement des colonies de fourmis (KATYAL & MISHRA, 2013 ; LI et al., 2011 ; LIU et al., 2006) ou celui des abeilles butineuses (KATYAL & MISHRA, 2013 ; RANDLES et al., 2010 ; SESUM-CAVIC & KÜHN, 2010a, 2010b), permettent aux ressources de prendre des décisions localement, tout

en contribuant à un équilibrage émergent du système. Les stratégies multi-agents, où chaque ressource agit en fonction de son propre apprentissage et de son interaction avec ses voisines, offrent notamment une prise de décision adaptative et dynamique.

L'avantage principal de ces méthodes réside dans leur capacité d'adaptation aux variations imprévues du système, permettant un équilibrage efficace même en conditions incertaines. Cependant, elles impliquent un coût computationnel plus élevé que les méthodes basées sur des règles, et nécessitent parfois une phase d'entraînement avant d'atteindre une performance optimale. En outre, les différentes approches décisionnelles peuvent être mélangées : une méthode basée sur des règles peut être couplée à de l'apprentissage pour modifier ses paramètres d'exécution selon l'état du système par exemple (seuils, critères d'optimalité des ressources, etc.).

2.2 Métriques de performance

L'équilibrage de charge joue un rôle fondamental dans l'optimisation des performances des systèmes informatiques. Qu'il s'agisse de serveurs, d'architectures cloud, de systèmes embarqués ou de composants électroniques, l'efficacité d'un mécanisme d'équilibrage repose sur sa capacité à répartir la charge de manière optimale, en tenant compte de divers critères de performance. Chaque stratégie d'équilibrage de charge possède des objectifs spécifiques qui influencent son adoption dans les systèmes nécessitant un tel mécanisme.

Pour évaluer la pertinence et l'efficacité d'une stratégie d'équilibrage, plusieurs métriques de performance sont utilisées. Ces métriques permettent de mesurer la rapidité d'exécution des requêtes, la charge des ressources, l'équité de répartition ou encore la robustesse du système face aux variations de charge. À ces aspects classiques s'ajoute aujourd'hui une préoccupation croissante pour l'efficacité énergétique, notamment dans le contexte des centres de données et des systèmes à grande échelle.

Cette section présente les principales métriques utilisées pour évaluer les systèmes d'équilibrage de charge (BELGAUM et al., 2020 ; JADER et al., 2019 ; MISHRA et al., 2020 ; ROY et al., 2019), que l'on peut regrouper en deux catégories complémentaires : métriques classiques et spécifiques à l'équilibrage de charge. Nous examinerons d'abord les métriques classiques, telles que le temps de réponse, le débit et l'utilisation des ressources. Ces indicateurs, couramment employés pour évaluer la performance de tout système informatique, permettent de mesurer son efficacité globale, indépendamment de l'intégration d'un mécanisme d'équilibrage de charge. Nous aborderons ensuite les métriques spécifiques à l'équilibrage, comme l'équité de répartition, le taux de migration de charge et la scalabilité. Celles-ci sont conçues pour analyser la manière dont une solution d'équilibrage influence la distribution de la charge et l'adaptabilité du système face aux variations de demande. Un bon mécanisme d'équilibrage doit non seulement améliorer les métriques classiques (en réduisant le temps de réponse et en optimisant l'utilisation des ressources), mais aussi limiter les effets négatifs mesurés par les métriques spécifiques, tels qu'un nombre excessif de migrations de charge ou la surcharge de certains nœuds. Enfin, nous nous intéresserons aux métriques de robustesse, qui occupent une place privilégiée dans nos travaux. Ces indicateurs permettent d'évaluer la capacité d'un système à faire face aux pannes et aux pics de charge, des enjeux cruciaux pour les infrastructures modernes, où la continuité de service et la stabilité sont primordiales.

2.2.1 Métriques classiques

L'évaluation des performances d'un système informatique repose sur des mesures fondamentales permettant d'analyser son efficacité globale. Ces métriques classiques sont indépendantes des mécanismes d'équilibrage de charge et sont utilisées pour mesurer la réactivité, la capacité de traitement et l'exploitation des ressources d'un système, qu'il soit centralisé ou distribué.

Dans le contexte de l'équilibrage de charge, ces indicateurs restent essentiels, car un bon mécanisme d'équilibrage doit non seulement répartir efficacement la charge, mais aussi améliorer les performances globales du système. Un temps de réponse réduit, un débit élevé et une utilisation optimale des ressources sont autant de critères qui garantissent un fonctionnement fluide et performant. Ces aspects influencent directement l'expérience utilisateur et les temps de calcul, en assurant des interactions rapides, une latence minimale et une stabilité accrue du service.

Cette section examine les principales métriques classiques. Nous commencerons par le temps de réponse, qui reflète la rapidité du système à traiter les requêtes, avant d'aborder le débit, indicateur clé de sa capacité de traitement. Nous nous pencherons ensuite sur l'utilisation des ressources, qui permet d'évaluer l'efficacité d'exploitation des composants matériels. Enfin, nous étudierons la consommation énergétique, un critère directement lié à l'utilisation des ressources, qui est de plus en plus crucial dans l'optimisation des infrastructures modernes.

2.2.1.1 Temps de réponse

Le temps de réponse, aussi qualifié de latence du système ou de qualité de service (QoS), correspond au temps total que met le système à fournir un résultat lors d'une requête. Cette métrique est particulièrement importante pour le web. Jakob Nielsen a partagé dans le chapitre 3 de son livre *Usability engineering* (1993) trois limites principales de temps de réponse (tr) pour les interfaces utilisateur (NIELSEN, 1993) :

- $tr \leq 0{,}1$ seconde : réponse perçue comme instantanée ;
- $0{,}1 < tr \leq 1$ seconde : perte de la sensation d'instantanéité, mais le flux de pensée de l'utilisateur reste ininterrompu ;
- $1 < tr \leq 10$ secondes : attente encore acceptable, mais nécessite un indicateur visuel ; perte de l'attention de l'utilisateur au-delà.

De nos jours, les utilisateurs ont tendance à abandonner la navigation d'un site web présentant un temps de réponse de plus de 3 secondes pour afficher une page. Google conseille un temps de réponse maximum de 2,5 secondes pour l'affichage du contenu principal, peu importe l'appareil utilisé (« Largest Contentful Paint | Lighthouse », 2020), sans quoi la perte d'utilisateurs due à l'attente augmente fortement. Cette limite de rétention tend à diminuer avec le temps, en adéquation avec la recherche d'instantanéité par la société et la réduction du niveau de concentration des nouvelles générations.

Le temps de réponse englobe plusieurs métriques plus précises qui interviennent de manière successive tout au long du processus d'interaction avec le système. En premier lieu, le temps de transmission de la requête, qui comprend la latence du réseau, le processus de sélection du serveur par le système d'équilibrage de charge

(dont l'efficacité dépend de la stratégie adoptée) ainsi que le délai d'attente avant traitement, lequel est influencé par la charge et la capacité du serveur choisi. Cette première métrique souligne l'importance d'un système d'équilibrage de charge performant, car celui-ci influence directement le temps de réponse global du système. Ensuite, le temps consacré au traitement de la requête par le serveur, qui inclut le temps de lecture des données, est pris en compte. Enfin, la latence associée à la communication de la réponse est également intégrée.

2.2.1.2 Débit

Le débit est une mesure clé de la performance d'un système, représentant la quantité de travail achevé par unité de temps. Exprimé sous la forme $D = \frac{N}{T}$, où N est la quantité de travail achevé (exprimé en nombre de tâches, requêtes, opérations, etc.) et T la période d'observation (généralement en secondes), le débit reflète la capacité du système à absorber une charge de travail donnée.

Le débit et le temps de réponse sont étroitement liés, puisque le second influence directement le premier. Une réponse rapide permet de pouvoir traiter plus de tâches dans un laps de temps donné. Toutefois, un débit élevé n'implique pas nécessairement un temps de réponse faible. Un système peut traiter un grand nombre de tâches tout en ayant des temps de réponse longs si celles-ci sont mises en attente ou s'il y a de la latence de communication par exemple. Un bon mécanisme d'équilibrage de charge vise donc à maximiser le débit tout en maintenant un temps de réponse faible en exploitant pleinement la capacité disponible du système par une répartition efficace de la charge.

2.2.1.3 Utilisation des ressources physiques

L'utilisation des ressources, au centre des performances des systèmes informatiques, correspond à la proportion (généralement exprimée en pourcentage) dans laquelle un système exploite ses ressources matérielles et logicielles (processeur, mémoire, réseau, stockage, etc.). Cette métrique permet d'évaluer l'efficacité de l'allocation des ressources et d'identifier les éventuels goulets d'étranglement pour limiter leur impact sur les performances du système.

Une utilisation optimale des ressources signifie qu'elles sont exploitées efficacement sans être sous-utilisées (potentiel gaspillé, inefficacité énergétique) ni surchargées (dégradation des performances, augmentation du temps de réponse). Dans un système bien équilibré, l'utilisation des ressources doit être homogène pour éviter qu'un goulet d'étranglement n'affecte ses performances, et donc la qualité de service.

2.2.1.4 Consommation énergétique

La consommation énergétique des infrastructures et services cloud constitue un enjeu majeur du monde numérique moderne. Avec la multiplication des services et la demande croissante en puissance de calcul, la consommation énergétique des centres de données et systèmes distribués augmente chaque année, entraînant une hausse des coûts d'exploitation et une empreinte carbone accrue. Cette tendance soulève des défis à la fois économiques et environnementaux, nécessitant des solutions efficaces pour limiter l'impact énergétique des systèmes informatiques.

La consommation d'énergie est généralement exprimée en joules (J) ou en watts-heures (Wh). Une autre métrique couramment utilisée est la puissance moyenne consommée (W) sur une période donnée. Dans le contexte des centres de données et systèmes distribués, la consommation d'énergie est souvent évaluée via des indicateurs comme l'efficacité d'utilisation de l'énergie (MALONE & BELADY, 2006) (Power Usage Effectiveness, PUE), qui mesure l'efficacité énergétique globale d'un centre de données. La mesure est un ratio entre la consommation totale de l'infrastructure informatique du centre (serveurs, stockage et conversion de l'énergie, systèmes de refroidissement) et uniquement celle utilisée par ses serveurs : PUE $= \frac{\text{énergie totale}}{\text{énergie serveurs}}$. Une valeur proche de 1 signifie une utilisation efficace de l'énergie.

Une autre métrique couramment utilisée pour évaluer l'efficacité énergétique des systèmes est le produit énergie-délai (Energy-Delay Product, EDP) (GONZALEZ & HOROWITZ, 1996). Celui-ci établit un compromis entre la consommation énergétique et le temps de réponse, permettant ainsi d'évaluer l'impact des optimisations sur ces deux aspects. Le produit énergie-délai est défini comme suit : $EDP(J \cdot s) = E(J) \times D(s)$, où $E(J)$ est la consommation d'énergie en joules et $D(s)$ le temps de réponse en secondes. Une valeur faible du produit est souhaitable, car elle indique un compromis optimal entre efficacité énergétique et temps de réponse. Une diminution de la valeur indique une amélioration du système, soit en réduisant sa consommation d'énergie à délai constant, soit en accélérant l'exécution sans surcoût énergétique, soit en améliorant les deux facteurs à la fois. À l'inverse, une augmentation de la valeur traduit une détérioration des performances, impliquant soit une consommation excessive, soit un allongement du temps de réponse, soit les deux. Lorsque le produit reste constant malgré des ajustements, cela signifie que les gains réalisés sur un facteur sont compensés par une dégradation équivalente de l'autre.

De nombreux travaux de recherche, tels que HASAN et al. (2017), KATAL et al. (2023), LEE et ZOMAYA (2012), LIN et al. (2018) et UCHECHUKWU et al. (2014), se focalisent sur la modélisation de la consommation énergétique des infrastructures cloud afin d'identifier les principaux leviers d'optimisation. Étant étroitement liée à l'utilisation des ressources, la consommation énergétique peut être réduite grâce à des améliorations à la fois matérielles et logicielles. Les optimisations matérielles visent à améliorer l'efficacité énergétique des composants en ajustant dynamiquement la tension et la fréquence des processeurs (Dynamic Voltage and Frequency Scaling), à optimiser la gestion de la mémoire et du stockage (réduction des accès en lecture/écriture), et à adopter des systèmes de refroidissement plus efficients pour limiter la dissipation thermique. Les optimisations logicielles, quant à elles, interviennent à plusieurs niveaux, notamment à travers la virtualisation des serveurs, qui permet une meilleure mutualisation des ressources, l'optimisation des programmes pour réduire leur demande en calcul et en mémoire, ainsi que l'intégration d'algorithmes d'équilibrage de charge visant à répartir intelligemment la charge tout en minimisant la consommation énergétique. L'objectif est de trouver un compromis entre performance, qualité de service et efficacité énergétique. Par ailleurs, l'utilisation d'énergies renouvelables constitue un levier supplémentaire dans cette démarche. Des entreprises comme Google se sont engagées dans cette voie, avec l'ambition d'alimenter leurs centres de données exclusivement en énergie verte d'ici 2030 (« Un fonctionnement écoresponsable - Centres de données Google », s. d.).

Dans ce contexte, les mécanismes d'équilibrage de charge jouent un rôle clé en répartissant intelligemment

la charge de travail pour éviter la surcharge de certains nœuds, réduire la consommation énergétique globale et améliorer l'efficacité des infrastructures. Une approche bien conçue permet ainsi de trouver un compromis entre performance, coûts et impact environnemental.

2.2.2 Métriques spécifiques

Si les métriques classiques (temps de réponse, débit, utilisation des ressources, etc.) permettent d'évaluer la performance globale d'un système, elles ne suffisent pas à mesurer l'efficacité des mécanismes d'équilibrage de charge. En effet, un bon équilibrage ne se limite pas à garantir de bonnes performances en termes de rapidité et d'efficacité, mais il doit aussi assurer une répartition optimale de la charge, une gestion efficace des migrations de tâches et une scalabilité performante.

Dans cette section, nous introduisons des métriques spécifiques à l'équilibrage de charge, conçues pour évaluer les performances des algorithmes et stratégies de répartition. Nous commencerons par l'équité de répartition de la charge, qui correspond au degré d'uniformité dans la distribution des tâches entre les ressources. Nous continuerons avec le taux de migration de charge, qui quantifie la fréquence et le coût des transferts de charge entre les ressources. Enfin, nous nous intéresserons à la scalabilité, qui évalue la capacité du système à s'adapter efficacement à l'augmentation de la charge, ou à l'ajout de nouvelles ressources sans dégradation des performances.

2.2.2.1 Équité de répartition de la charge

L'équité de répartition de la charge est une métrique qui sert à mesurer le degré d'uniformité dans la distribution du travail entre les différentes ressources du système. L'objectif derrière l'utilisation de cette métrique est de détecter un déséquilibre dans l'utilisation des ressources pour identifier une surcharge (dégradation des performances et allongement du temps de réponse) ou une sous-utilisation (gaspillage des capacités disponibles) de certaines ressources. Un système d'équilibrage de charge efficace doit tendre vers une distribution équitable afin d'optimiser à la fois les performances et l'utilisation des ressources.

La méthode de mesure d'équité la plus courante est l'indice de Jain (JAIN et al., 1984), qui propose une mesure répondant mieux à l'objectif (évaluer l'équité de distribution) que des mesures plus classiques comme la variance ou le coefficient de variation ($\frac{\text{variance}}{\text{moyenne}}$). L'indice propose une valeur allant de 0 à 1 et se calcule de la manière suivante :

$$J = \frac{(\sum_{i=1}^{N} L_i)^2}{N \sum_{i=1}^{N} L_i^2} \tag{2.1}$$

où N est le nombre de ressources et L_i la charge (**L**oad) de la ressource i et $J \in [\frac{1}{N}, 1]$. Lorsque J est proche de 0, la distribution de la charge est fortement déséquilibrée, tandis que lorsque $J = 1$, la charge est répartie de manière parfaitement équitable. Cependant, l'indice a pour but d'être utilisé en environnement homogène (ressources uniformes) ; il ne prend pas en compte les potentielles variances de capacités entre les ressources.

Prenons l'exemple de deux ressources A et B pour lesquelles $C_A = 4$, $C_B = 2$ et $L_A = L_B = 3$: l'indice de Jain aurait une valeur de 1 (égalité parfaite du nombre de tâches entre les ressources), bien que la ressource B soit chargée à 150% de sa capacité et la ressource A ne soit qu'à 75%.

L'indice peut toutefois être adapté en pondérant la charge des ressources par leur capacité comme suit :

$$J_p = \frac{(\sum_{i=1}^{N} \frac{L_i}{C_i})^2}{N \sum_{i=1}^{N} (\frac{L_i}{C_i})^2} \tag{2.2}$$

où C_i est la capacité de la ressource i. L'indice indique alors une équité parfaite lorsque la proportion de charge par rapport à la capacité est identique pour toutes les ressources. On peut différencier les deux indices : la forme initiale mesure l'égalité de la répartition (charge identique sur toutes les ressources), tandis que la forme pondérée en mesure réellement l'équité (charge proportionnelle à la capacité des ressources). Si l'on reprend l'exemple précédent, l'indice est alors $J_p = 0{,}9$.

Le Tableau 2.2 propose une série de scénarios de répartitions de charge avec les deux indices (d'égalité et d'équité) calculés. On peut voir que même si l'indice de Jain initial indique une équité parfaite, sa forme pondérée en est très éloignée dans certains cas (scénario 3). À l'inverse, lorsque l'équité est parfaite proportionnellement aux capacités, l'indice non pondéré ne l'indique pas forcément (scénario 5).

2.2.2.2 Migration de la charge

La migration de la charge correspond à la redistribution dynamique des tâches entre les ressources d'un système. Elle ne concerne que les systèmes présentant de l'adaptabilité ; les algorithmes tels que le Round Robin en sont complètement dépourvus par exemple. Plusieurs métriques permettent d'évaluer l'efficacité de cette migration : le nombre de migrations, et le temps et le coût d'une migration. Cet ensemble de métriques permet de mesurer l'efficacité du système à répartir efficacement les tâches pour s'adapter à la charge.

Le nombre de migrations correspond à la longueur du trajet des tâches (nombre de sauts d'une ressource à une autre) dans le système pour trouver leur ressource de traitement. Un trajet long indique que le système peine à trouver une destination pour la tâche, impliquant une dépense énergétique plus élevée et un temps de réponse allongé.

Le temps de migration quantifie la durée nécessaire pour transférer une tâche d'une ressource à une autre. Dans certaines études il correspond à une étape de migration, tandis pour d'autres la métrique correspond au temps total de transfert d'une tâche vers une ressource adéquat (possibles migrations multiples). Les migrations pouvant survenir au niveau des instances virtuelles (grâce à la virtualisation des serveurs) mais également au niveau physique (entre machines), le temps de migration dépend de plusieurs facteurs, tels que la taille des données à transférer et la latence réseau. Un temps de migration trop long peut retarder l'exécution des tâches et impacter les performances globales du système, notamment son temps de réponse.

Le coût de migration mesure les coûts computationnels et communicationnels d'une migration entre deux

Scénario 1 : charge = capacité = 1												
Ressource	1	2	3	4	5	6	7	8	9	10	**Indice**	**Indice pondéré**
Charge	1	1	1	1	1	1	1	1	1	1	1	1
Capacité	1	1	1	1	1	1	1	1	1	1		

Scénario 2 : charge fixe faible, capacité simple et double												
Ressource	1	2	3	4	5	6	7	8	9	10	**Indice**	**Indice pondéré**
Charge	1	1	1	1	1	1	1	1	1	1	1	0,9
Capacité	1	1	1	1	1	2	2	2	2	2		

Scénario 3 : charge fixe élevée, capacité simple et quintuple												
Ressource	1	2	3	4	5	6	7	8	9	10	**Indice**	**Indice pondéré**
Charge	5	5	5	5	5	5	5	5	5	5	1	$\simeq 0{,}69$
Capacité	1	1	1	1	1	5	5	5	5	5		

Scénario 4 : charge et capacité inversées												
Ressource	1	2	3	4	5	6	7	8	9	10	**Indice**	**Indice pondéré**
Charge	3	3	3	2	2	2	1	1	1	0	$\simeq 0{,}77$	$\simeq 0{,}56$
Capacité	1	1	1	2	2	2	3	3	3	4		

Scénario 5 : charge = capacité												
Ressource	1	2	3	4	5	6	7	8	9	10	**Indice**	**Indice pondéré**
Charge	1	2	3	4	5	6	7	8	9	10	$\simeq 0{,}79$	1
Capacité	1	2	3	4	5	6	7	8	9	10		

TABLE 2.2 – Exemples de scénarios de distribution de charge sur 10 ressources. Pour chaque exemple, l'indice de Jain et sa version pondérée sont proposés afin de mettre en évidence l'apport de précision de la pondération.

ressources du système. Les coûts sont généralement quantifiés en termes de temps, d'impact sur les performances des ressources (utilisation supplémentaire du processeur et de la mémoire) et d'énergie consommée pour réaliser la migration. Selon la quantification, la mesure est impactée par divers facteurs tels que la latence réseau, la taille des données à lire, écrire et communiquer, ou encore la charge actuelle des ressources.

2.2.2.3 Scalabilité et adaptabilité

La scalabilité et l'adaptabilité sont deux métriques qui permettent de mesurer la capacité du système d'équilibrage à faire face à des changements dans son environnement. La scalabilité se concentre sur la mesure du passage à l'échelle de la charge ou du nombre de ressources, tandis que l'adaptabilité quantifie la manière dont le système réagit à une surcharge ou à une panne par exemple. Il n'existe pas de manière précise de calculer ces métriques comme les précédentes. Elles s'évaluent plutôt par l'analyse de l'évolution d'autres métriques telles que le débit, le temps de réponse et l'équité de répartition.

Pour mesurer la scalabilité, il est par exemple possible de mesurer la différence du débit et du temps de réponse entre avant et après une mise à l'échelle de l'environnement (augmentation conséquente de la charge ou des ressources). L'objectif est alors d'avoir un débit qui corresponde au nouvel environnement et de conserver un temps de réponse faible. Un système d'équilibrage qui n'est pas scalable peinera à dispatcher la charge

efficacement, ce qui augmentera le temps de réponse.

L'adaptabilité, quant à elle, peut être mesurée par le temps que prend le système à retrouver un état d'équilibre, après une perturbation (panne d'une ressource ou surcharge). Elle peut notamment être quantifiée par l'évolution de l'équité de la répartition de la charge entre avant et après la perturbation. L'objectif des systèmes d'équilibrage adaptatifs est de réagir rapidement et efficacement aux événements inattendus qui peuvent survenir.

Un bon système d'équilibrage doit être scalable et adaptable afin de fournir de bonnes performances peu importe la situation. Ces métriques sont notamment directement liés aux concept de robustesse et de résilience.

2.2.3 La robustesse

La robustesse est une métrique essentielle pour évaluer la capacité d'un système d'équilibrage de charge à maintenir ses performances face à des perturbations, des pannes ou des variations imprévues de la charge. Un système robuste doit être capable de continuer à fonctionner efficacement même en cas de défaillance de certaines ressources ou de pics de demande. Cette section explore les différentes dimensions de la robustesse et les possibilités pour l'évaluer.

2.2.3.1 Définition

De manière générale, la robustesse désigne la capacité d'un système à préserver ses fonctions ou caractéristiques essentielles malgré des perturbations internes ou externes. Cette propriété se manifeste dans de nombreux domaines, des systèmes biologiques aux infrastructures d'ingénierie, en passant par les réseaux complexes.

En biologie, la robustesse constitue une propriété fondamentale des systèmes évolutifs complexes (KITANO, 2004). Elle permet aux organismes de survivre dans des environnements imprévisibles et de fonctionner malgré des constituants potentiellement défaillants. Cette robustesse ne signifie pas l'immuabilité du système, mais sa capacité à maintenir certaines fonctions clés en s'adaptant de manière flexible. Elle peut se traduire par un retour à un état stable existant (attracteur) ou par une transition vers un nouvel attracteur préservant les fonctions essentielles, ce qui s'apparente à une forme de résilience.

Dans les systèmes informatiques et d'ingénierie, la robustesse correspond à la capacité de maintenir un fonctionnement correct sur un large éventail de conditions, tout en dégradant ses performances de manière contrôlée au-delà de ces limites (GRIBBLE, 2001). Les approches basées sur la prévision exhaustive des scénarios sont remises en question, car elles peuvent introduire une certaine fragilité par l'omission involontaire de scénarios.

La théorie de la Tolérance Hautement Optimisée (HOT) (CARLSON & DOYLE, 2002) montre que les systèmes complexes sont souvent conçus pour être très robustes face aux perturbations fréquentes, mais vulnérables à celles qui sont rares (fragilité). Cette robustesse sélective repose souvent sur une grande complexité interne, ce qui accroît la difficulté de gestion des risques.

Un compromis émerge alors entre robustesse, fragilité, performance et consommation de ressources. Un système peut se révéler robuste dans certains cas, tout en étant extrêmement vulnérable dans d'autres. Com-

prendre ces compromis est crucial pour localiser les points de défaillance potentiels et mettre en œuvre des mécanismes de protection efficaces.

Dans le cas de l'équilibrage de charge, la robustesse se manifeste par la capacité du système à absorber des hausses de charge imprévues (perturbations externes) ou des défaillances de ressources (perturbations internes). Le système doit pouvoir encaisser une surcharge momentanée et revenir à un état stable. Lorsqu'une ressource devient indisponible, la distribution de la charge doit être ajustée pour limiter l'impact, au moins jusqu'à sa remise en service. Toutefois, une surcharge extrême ou un nombre important de défaillances peut entraîner une saturation ou un effondrement du système. On retrouve ici la dualité robustesse/fragilité.

2.2.3.2 Évaluation

La robustesse d'un système s'évalue généralement en comparant un indice de performance avant et après une perturbation. Un système est dit robuste si cet indice varie peu, traduisant une stabilité fonctionnelle malgré les perturbations. À l'inverse, une variation significative révèle une incapacité à s'adapter efficacement.

Dans les travaux de HU et al. (2021), la robustesse est étudiée dans le contexte des réseaux électriques. Les auteurs proposent un indice évaluant l'efficacité du réseau à acheminer l'électricité des nœuds générateurs vers les autres. Cet indice repose notamment sur la centralité d'intermédiarité électrique (electricity betweenness centrality), une extension de la centralité d'intermédiarité classique (NEWMAN, 2005), qui prend en compte l'ensemble des chemins possibles plutôt que seulement les plus courts. En simulant divers scénarios de défaillance, les auteurs comparent les valeurs de l'indice avant et après perturbation pour en déduire la robustesse du système.

Dans les systèmes complexes fondés sur une structure en réseau, la robustesse est étroitement liée à l'architecture du graphe. Si la défaillance d'un seul nœud entraîne une dégradation marquée de la connectivité – et donc des performances –, le système est jugé peu robuste. Des indicateurs tels qu'une forte connectivité, un coefficient de clustering élevé, une faible longueur moyenne des plus courts chemins et un petit diamètre sont généralement associés à une meilleure robustesse (ZHANG et al., 2015). Ainsi, dans les études de DEKKER et COLBERT (2004) et ZHANG et al. (2015), la robustesse est mesurée en évaluant combien de nœuds peuvent être retirés avant que le réseau ne se fragmente en sous-graphes disjoints.

Dans le domaine informatique, la fiabilité du matériel et du logiciel constitue également un indicateur clé de robustesse. On utilise souvent des mesures telles que le temps moyen avant défaillance (Mean Time To Failure) et le temps moyen entre deux défaillances (Mean Time Between Failures) (KRASICH, 2009). Le temps de réparation ou de remplacement d'un composant défaillant (Mean Time To Repair) (MUHAMMAD RIDZUAN & DJOKIC, 2019) est également déterminant. Ainsi, un système subissant peu de défaillances et capable d'y remédier rapidement présente un niveau de robustesse élevé.

2.3 Classification algorithmique

De nombreuses études, telles que KATYAL et MISHRA (2013), PATEL et al. (2016), JAFARNEJAD GHOMI et al. (2017), KUMAR et KUMAR (2019), POURGHEBLEH et HAYYOLALAM (2020), SHAFIQ et al. (2022) et KANELLOPOULOS et SHARMA (2022), proposent des analyses comparatives de diverses approches d'équilibrage de charge. Chacune adopte un angle particulier pour classer et évaluer les méthodes proposées.

Dans cette section, nous proposons une classification algorithmique fondée sur l'environnement d'exécution des solutions. La Figure 2.6 présente cette taxonomie sous forme d'un arbre, permettant une visualisation synthétique des grandes familles d'approches.

Nous commencerons par explorer les algorithmes conçus pour des environnements statiques et ne nécessitant pas d'adaptation, incluant les méthodes optimales et sous-optimales. Nous aborderons ensuite les environnements dynamiques nécessitant de l'adaptation, en distinguant les approches centralisées, semi-centralisées et décentralisées ; ces dernières pouvant intégrer ou non des mécanismes de coopération. Pour chacune de ces catégories, nous présenterons des exemples représentatifs afin d'illustrer les principes de fonctionnement propres à chaque famille de solutions.

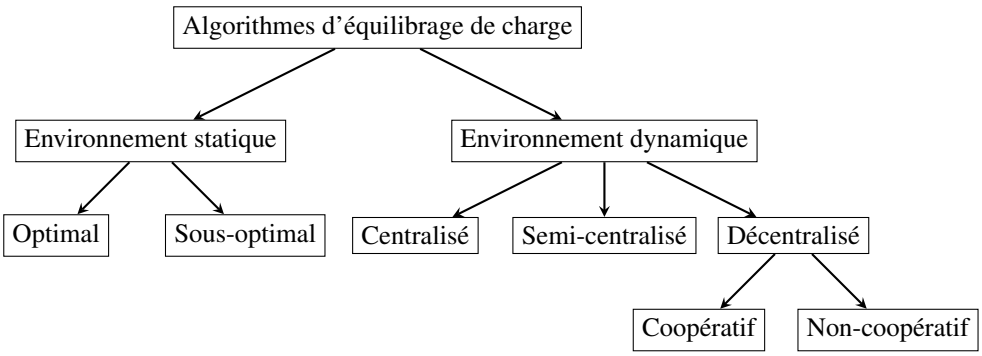


FIGURE 2.6 – Classification algorithmique des méthodes d'équilibrage selon le type d'environnement.

2.3.1 Équilibrage en environnement statique

Cette section est consacrée aux algorithmes conçus pour un environnement statique. Par définition, un tel environnement se caractérise par un nombre de ressources fixe et connu à l'avance, tout comme leur capacité de traitement. Dans certains cas, la charge de travail à répartir est elle aussi préalablement déterminée.

Dans ce cadre, plusieurs approches sont envisageables, que l'on peut regrouper en deux grandes catégories : les méthodes optimales et sous-optimales. Nous nous intéresserons d'abord aux solutions optimales, puis aux stratégies sous-optimales.

2.3.1.1 Optimalité

L'optimalité des algorithmes d'équilibrage de charge est définie relativement à un critère de performance spécifique, tel que la minimisation du temps de réponse, la maximisation du débit ou encore la réduction de

l'écart de charge entre les ressources. Un algorithme dit optimal est formellement conçu pour atteindre la meilleure performance possible selon le critère retenu.

Ces algorithmes reposent en général sur des méthodes exactes, c'est-à-dire capables de garantir une solution optimale au problème posé, notamment via de la programmation linéaire. Dans leurs travaux, PINAR et AYKANAT (2004) recensent et améliorent plusieurs de ces approches, parmi lesquelles on trouve notamment la programmation dynamique. Celle-ci consiste à décomposer le problème global en sous-problèmes plus simples, dont les solutions optimales sont ensuite combinées pour obtenir une solution globale.

PINAR et AYKANAT (2004) explorent également des méthodes fondées sur la recherche paramétrique. Ces approches visent à identifier les valeurs optimales de certains paramètres clés du système (comme la capacité des ressources ou la charge entrante) afin d'optimiser le comportement général. Enfin, les algorithmes d'amélioration itérative sont aussi étudiés : ces méthodes partent d'une solution initiale, souvent fournie par une heuristique, qu'elles affinent progressivement via des ajustements successifs.

L'algorithme proposé par ROSS et YAO (1991) illustre bien cette approche d'optimisation globale. Il traite à la fois l'équilibrage de la charge entre les ressources et l'ordonnancement des tâches au sein de chaque ressource. L'ordonnancement est modélisé comme un problème de polymatroïde, permettant de déterminer une fonction de délai moyen propre à chaque ressource. Sur cette base, la répartition de la charge est formulée comme un problème d'optimisation non linéaire, dont la résolution vise à minimiser le temps de réponse moyen du système.

De leur côté, TANTAWI et TOWSLEY (1985) s'intéressent également à la minimisation du temps de réponse moyen, tout en prenant en compte les délais de communication au sein du réseau. Ils proposent deux algorithmes. Le premier, fondé sur une étude paramétrique, analyse l'évolution de la solution optimale en fonction du temps de communication. Il examine notamment comment les rôles des nœuds (sources, puits, neutres) changent avec la variation de ce paramètre. Le second, appelé algorithme au point unique, détermine la stratégie optimale d'équilibrage pour un ensemble donné de caractéristiques système. Il classe les nœuds selon leurs propriétés de délai, identifie les groupes sources et puits, et calcule pour chacun la charge à allouer de manière optimale.

2.3.1.2 Sous-optimalité

La sous-optimalité désigne une classe d'algorithmes capables de produire des résultats satisfaisants, souvent proches de l'optimal, voire dans certains cas optimaux, mais sans garantie formelle d'optimalité. Cette catégorie englobe un large éventail d'approches, incluant les heuristiques, les métaheuristiques, ainsi que certaines techniques d'apprentissage automatique.

Dans ces approches, certaines bénéficient d'un encadrement théorique partiel : elles possèdent une borne d'approximation garantissant que la solution obtenue est située à une certaine distance de l'optimum. C'est notamment le cas des méthodes de List Scheduling (ARABNEJAD & BARBOSA, 2014), qui affectent les tâches à la ressource la moins chargée au moment de leur arrivée, avec ou sans tri préalable, ou encore des schémas d'approximation en temps polynomial (Polynomial Time Approximation Scheme) (CHEKURI & KHANNA,

2005).

Les heuristiques constituent une sous-famille importante d'algorithmes sous-optimaux. Elles reposent sur des règles prédéfinies simples et sont conçues pour offrir des solutions rapides, au détriment de la garantie de performance optimale. Leur atout principal réside dans leur efficacité computationnelle, ce qui les rend particulièrement adaptées aux contextes contraints en temps ou en ressources. En revanche, leur pertinence dépend fortement de la qualité de la modélisation du problème et de la connaissance préalable de l'environnement, nécessaire pour ajuster les règles de décision.

Parmi les heuristiques classiques, on peut citer le tourniquet (HIDAYAT et al., 2020), basé sur une répartition cyclique des tâches, les algorithmes Min-Min et Max-Min, qui priorisent respectivement les tâches les plus courtes ou les plus longues, l'affectation aléatoire (AZAR et al., 1994), et l'ordonnancement selon la charge actuelle (List Scheduling) (ARABNEJAD & BARBOSA, 2014), qui affecte chaque tâche à la ressource la moins sollicitée à l'instant de sa soumission.

Certaines heuristiques plus élaborées, bien que plus coûteuses en temps de calcul, permettent d'atteindre des performances nettement supérieures. Ces méthodes, généralement inspirées de la nature, appartiennent souvent aux métaheuristiques. C'est une classe d'algorithme d'optimisation générale pouvant être adaptée à n'importe quel problème d'optimisation. On peut notamment citer le recuit simulé (VAN LAARHOVEN & AARTS, 1987), inspiré du processus de refroidissement des métaux, les algorithmes génétiques (MIRJALILI, 2019), qui simulent les mécanismes de sélection naturelle, les approches par colonies de fourmis (BIRATTARI & DORIGO, 2000 ; BIRATTARI et al., 2002 ; BLUM, 2005 ; DORIGO et al., 2006) ou par comportement alimentaire des abeilles (PHAM et al., 2005 ; YUCE et al., 2013), basées sur le comportement collectif des insectes, ou l'optimisation par essaim de particules (KENNEDY & EBERHART, 1995 ; RAMEZANI et al., 2014).

Dans ce qui suit, nous présentons plusieurs de ces algorithmes. Nous commencerons par le tourniquet et ses variantes, avant d'aborder plus en détail les approches inspirées par les comportements collectifs, notamment ceux des colonies de fourmis et des abeilles.

Le tourniquet Le tourniquet (Round Robin) (HIDAYAT et al., 2020) est l'un des algorithmes d'équilibrage de charge utilisant une heuristique des plus simples. Il repose sur une répartition cyclique des tâches entre les ressources disponibles. Le répartiteur maintient une liste ordonnée des ressources et distribue successivement chaque nouvelle tâche à la ressource suivante dans la liste. Une fois la dernière ressource atteinte, l'algorithme reprend l'itération depuis le début de la liste, d'où le nom de "tourniquet". Ce mécanisme se base sur le principe du premier arrivé, premier servi.

La Figure 2.7 illustre ce fonctionnement. L'équilibreur reçoit un flux de tâches qu'il répartit entre trois ressources indexées. À chaque tâche assignée, l'index de la ressource courante est incrémenté ; lorsqu'il atteint la fin de la liste, il est réinitialisé à zéro.

Cette méthode est particulièrement adaptée à des environnements homogènes et statiques, où les ressources présentent des capacités similaires et où les tâches sont relativement uniformes. Elle suppose également que les caractéristiques de la charge soient connues à l'avance, permettant ainsi un dimensionnement adéquat du système. En revanche, dès que les conditions d'exécution deviennent hétérogènes, que ce soit au niveau des

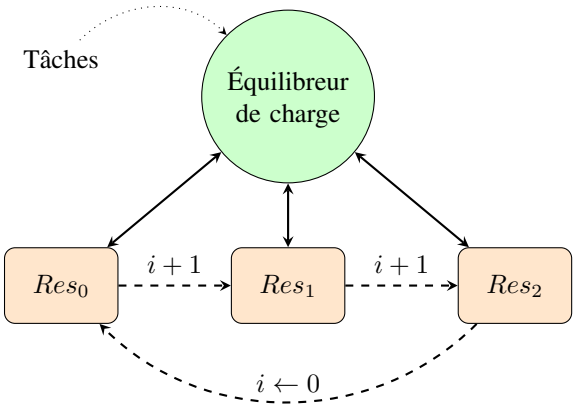


FIGURE 2.7 – Schématisation de l'algorithme du tourniquet. L'équilibreur affecte la tâche entrante à la ressource Res_i, i étant incrémenté à chaque affectation de tâche et remis à 0 lorsque toutes les ressources ont été parcourues.

ressources ou des tâches, l'algorithme montre ses limites. Ne tenant compte ni des performances des ressources ni des spécificités des tâches, il peut entraîner une surcharge de certains nœuds et une dégradation globale des performances.

Le tourniquet pondéré Le tourniquet pondéré (Weighted Round Robin) (DEVI & UTHARIARAJ, 2016) constitue une extension naturelle du tourniquet classique. Son principal apport réside dans la prise en compte des capacités relatives des ressources lors de la distribution des tâches au cours d'un tour. Le tourniquet classique peut ainsi être vu comme un cas particulier du tourniquet pondéré, dans lequel toutes les ressources possèdent une capacité identique.

Dans cette version pondérée, chaque ressource se voit attribuer un poids proportionnel à sa capacité, relativement à la somme des capacités de l'ensemble des ressources. Plus ce poids est élevé, plus la ressource recevra de tâches durant un tour. Si l'on reprend la schématisation du tourniquet de la Figure 2.7, une ressource Res_i est associée à un poids $p_i = c_i / \sum c_k$, où c_i est la capacité de la ressource. Par exemple, avec $c_0 = 1$, $c_1 = 2$ et $c_2 = 1$, les poids respectifs seront $p_0 = 0{,}25$, $p_1 = 0{,}5$ et $p_2 = 0{,}25$. Lors d'un tour, Res_1 recevra donc deux fois plus de tâches que Res_0 ou Res_2. Le tableau 2.3 illustre cette répartition, et la Figure 2.8 présente l'ordre d'attribution des tâches suivant : $Res_0 \rightarrow Res_1 \rightarrow Res_1 \rightarrow Res_2$.

i	c_i	p_i	Tâches par tour (tourniquet pondéré)	Tâches par tour (tourniquet)
0	1	0.25	1	1
1	2	0.5	2	1
2	1	0.25	1	1

TABLE 2.3 – Récapitulatif de l'exemple d'application de l'algorithme Weighted Round Robin avec comparaison à l'algorithme Round Robin.

Bien que cette méthode tienne compte des capacités pour répartir les charges de manière équitable, elle présente une inégalité temporelle dans la distribution : tant qu'une ressource n'a pas reçu l'intégralité des tâches qui lui reviennent selon son poids, l'algorithme continue de lui affecter des tâches sans passer à la ressource

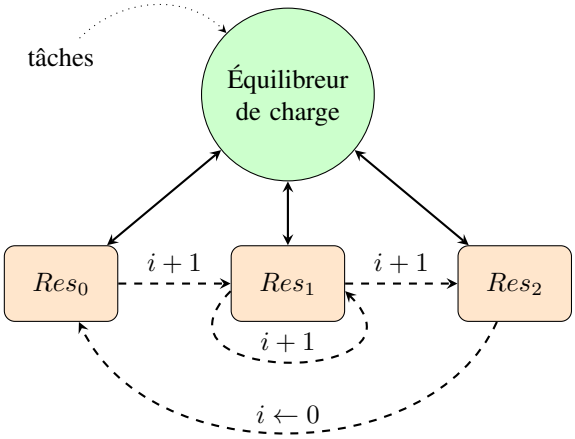


FIGURE 2.8 – Schématisation de l'exemple du tourniquet pondéré, où Res_1 reçoit deux fois plus de tâches que les autres ressources pendant le tour.

suivante. Ce comportement peut entraîner une augmentation du temps de réponse, notamment lorsque des tâches attendent d'être traitées par une ressource fortement pondérée alors qu'une autre, moins sollicitée, est disponible.

Pour répondre à cette limitation, KATEVENIS et al. (1991) ont introduit une variante appelée tourniquet pondéré entrelacé (Interleaved Weighted Round Robin), développée initialement dans un contexte matériel. L'idée principale est d'étaler l'attribution des tâches sur plusieurs cycles, au sein d'un même tour, afin d'assurer une meilleure répartition temporelle. Le nombre de cycles d'un tour correspond au maximum de tâches à attribuer à une ressource. Lors du cycle n, l'algorithme ne considère que les ressources devant recevoir au moins n tâches. Ainsi, toutes les ressources sont parcourues au premier cycle, seules celles ayant un poids supérieur au deuxième cycle, et ainsi de suite.

La Figure 2.9 illustre ce fonctionnement avec quatre ressources, dont Res_1 et Res_3 reçoivent deux fois plus de tâches que les autres. Le tour complet se compose de deux cycles successifs : $Res_0 \rightarrow Res_1 \rightarrow Res_2 \rightarrow Res_3$, puis $Res_1 \rightarrow Res_3$. L'étude de TABATABAEE et al. (2021) démontre une amélioration significative du temps de réponse par rapport au tourniquet pondéré classique.

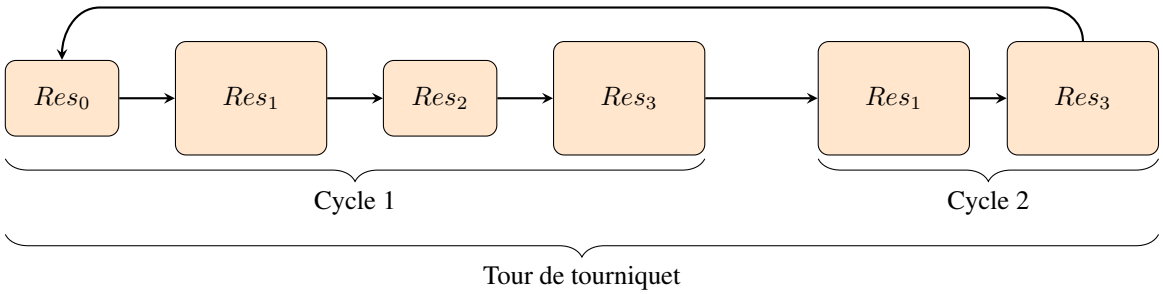


FIGURE 2.9 – Exemple du parcours de quatre ressources par l'algorithme du tourniquet pondéré entrelacé, où Res_1 et Res_3 attendent deux fois plus de tâches que Res_0 et Res_2.

Comme son prédécesseur, le tourniquet pondéré reste adapté à des environnements statiques. Toutefois,

il s'avère plus pertinent lorsque les ressources sont hétérogènes, car il répartit les charges en fonction des capacités. Néanmoins, cet algorithme ne prend toujours pas en compte les longueurs des tâches ni les priorités d'exécution. Bien qu'il constitue une amélioration notable du tourniquet de base, il demeure relativement limité face à des scénarios plus complexes.

Optimisation par colonies de fourmis L'optimisation par colonies de fourmis (Ant Colony Optimization) (BIRATTARI & DORIGO, 2000 ; BIRATTARI et al., 2002 ; BLUM, 2005 ; DORIGO et al., 2006) est une métaheuristique bio-inspirée conçue pour résoudre des problèmes d'optimisation combinatoire. Elle s'inspire du comportement collectif observé chez les colonies de fourmis lors de la recherche de nourriture. Dans la nature, ces insectes laissent derrière eux des traces de phéromones qui guident leurs congénères vers les sources d'alimentation. Les chemins les plus courts, parcourus plus fréquemment, voient leur concentration en phéromones augmenter, rendant ces trajets plus attractifs. Ce phénomène de communication indirecte, appelé stigmergie (DORIGO et al., 2000 ; THERAULAZ & BONABEAU, 1999), conduit à l'émergence collective de solutions satisfaisantes, voire quasi-optimales.

Pour appliquer cette métaheuristique à un problème donné, il est nécessaire de le modéliser sous forme d'un espace de recherche défini par les paramètres d'une fonction objectif à minimiser. L'objectif de l'algorithme est de découvrir un ensemble de valeurs des paramètres menant à une solution satisfaisante.

L'espace de recherche est exploré par des fourmis artificielles à travers une structure composée de trois graphes superposés :

— Le **graphe de construction** G_c est un graphe complet dont les sommets correspondent à toutes les valeurs possibles pour chaque paramètre.

— À partir de G_c, on construit le **graphe d'état** G, qui représente l'ensemble des solutions (partielles ou complètes). Les sommets de G sont les solutions, tandis que les arêtes définissent les règles de transition entre elles.

— Enfin, une fonction de représentation r permet de générer un **graphe de représentation** G_r, abstraction de G, dans lequel chaque sommet –appelé phantasme– constitue une perception simplifiée d'un état de G. Comme r est généralement non bijective, plusieurs états peuvent partager un même phantasme, ce qui réduit significativement la taille du graphe perçu par les fourmis.

Ainsi, les fourmis interagissent avec l'environnement perçu via G_r, où elles déposent des phéromones, tout en évoluant réellement dans G pour assurer la construction de solutions réalisables.

L'algorithme commence par l'application d'un niveau initial de phéromones τ_0 sur tous les sommets de G_r. Ce paramètre, réglé de manière empirique, joue un rôle crucial dans le comportement de l'algorithme : une valeur trop élevée peut entraîner une convergence prématurée vers une solution sous-optimale, tandis qu'une valeur trop faible ralentit l'exploration.

La méthode est un algorithme itératif qui s'exécute jusqu'à ce qu'une condition d'arrêt soit atteinte (nombre maximal d'itérations, stagnation, ou convergence). Chaque itération se décompose en deux phases principales :

1. **Construction des solutions :** un ensemble de fourmis est déployé dans G_r. Elles construisent incrémentalement des solutions complètes en se déplaçant de sommet en sommet. Le choix de la prochaine

étape est effectué de manière stochastique, influencé à la fois par la quantité de phéromones présentes et par des informations heuristiques spécifiques au problème (par exemple, une estimation de la qualité d'un choix local).

2. **Mise à jour des phéromones :** après que les fourmis ont construit leurs solutions, le graphe subit une double mise à jour :
   — Une évaporation des phéromones existantes, contrôlée par un taux ρ, permet d'éviter une domination trop rapide de certains chemins et encourage l'exploration continue.
   — Une augmentation des phéromones est ensuite réalisée en ajoutant une quantité Q de nouvelles phéromones, proportionnelle à la qualité des solutions produites. Cette étape renforce les trajectoires prometteuses et guide les fourmis futures vers les zones potentiellement optimales.

Ce double mécanisme permet d'équilibrer exploration et exploitation, et constitue le principe central de la stigmergie algorithmique. Il permet à la colonie de converger vers des solutions efficaces sans supervision centralisée.

Algorithme des abeilles L'algorithme des abeilles (Bees Algorithm) (PHAM et al., 2005 ; YUCE et al., 2013) est, tout comme l'approche par colonies de fourmis, une métaheuristique bio-inspirée issue de la famille de l'intelligence en essaim. Il s'inspire du comportement collectif des abeilles à la recherche de nectar, dans lequel certaines abeilles exploratrices prospectent l'environnement tandis que d'autres exploitent les zones les plus prometteuses signalées par leurs congénères.

Le principe repose sur une analogie entre l'espace de recherche et un environnement floral à butiner. Chaque point de cet espace représente une combinaison de paramètres de la fonction objectif à optimiser. Le but de l'algorithme est de découvrir progressivement les zones les plus "riches en nectar", c'est-à-dire celles qui fournissent les meilleures solutions à la fonction objectif.

L'algorithme fonctionne de manière itérative, avec un ensemble fixe d'abeilles réparties entre deux types de recherche : la recherche globale, pour explorer de nouveaux espaces, et la recherche locale, pour exploiter les zones déjà identifiées comme prometteuses.

Au début du processus, n sites de recherche sont générés aléatoirement dans l'espace des solutions, chacun étant occupé par une abeille. Une évaluation de la qualité de chaque site est ensuite réalisée à l'aide de la fonction objectif. L'algorithme entre alors dans un cycle d'optimisation composé de quatre étapes principales :

1. **Sélection des sites prometteurs :** parmi les n sites visités, les m meilleurs sont retenus pour une exploration approfondie. Ces sites sont classés en deux groupes : sites élites (les plus prometteurs) et sites non-élites (ayant un bon potentiel, mais moindre).
2. **Répartition des abeilles :** des abeilles supplémentaires sont assignées aux sites sélectionnés pour mener une recherche locale. Les sites élites reçoivent un plus grand nombre d'abeilles afin de maximiser l'exploration dans les zones les plus prometteuses.
3. **Recherche locale :** les abeilles affectées à un site échantillonnent l'environnement immédiat en explorant légèrement autour de la solution actuelle (modification des paramètres). Parmi les nouvelles

solutions générées localement, la meilleure est retenue comme représentante du site, tandis que les autres abeilles sont supprimées (une dynamique inspirée du processus de sélection dans les algorithmes génétiques (MIRJALILI, 2019)). Si aucune solution meilleure n'est trouvée, le site est considéré comme un optimum local (potentiellement global) et retiré de la recherche.

4. **Recherche globale :** les abeilles restantes (non affectées à un site prometteur) effectuent une exploration aléatoire dans l'espace global, générant de nouveaux sites comme lors de l'initialisation.

Ce processus est répété jusqu'à ce qu'un critère d'arrêt soit atteint, par exemple un nombre d'itérations maximal ou une amélioration négligeable de la solution.

Grâce à l'alternance entre exploitation (recherche locale) et exploration (recherche globale), l'algorithme des abeilles parvient à converger vers des solutions de haute qualité tout en évitant de se figer dans des optima locaux.

2.3.2 Équilibrage en environnement dynamique

Cette section s'intéresse aux algorithmes conçus spécifiquement pour des environnements dynamiques, dans lesquels la charge de travail est à la fois inconnue a priori et sujette à des fluctuations importantes dans le temps, tout comme le nombre et les capacités des ressources disponibles. De telles caractéristiques rendent les approches statiques inefficaces et exigent le recours à des stratégies d'équilibrage de charge adaptatives, capables de réagir en temps réel aux évolutions du système.

Les solutions présentées sont classées selon leur architecture de prise de décision. Une distinction est opérée entre les approches centralisées, semi-centralisées et décentralisées. Ces dernières sont à leur tour subdivisées en deux catégories : les approches coopératives, dans lesquelles les entités échangent des informations pour coordonner leur comportement, et les approches non coopératives, où chaque entité agit de manière autonome, sans coordination explicite avec les autres.

Nous commencerons par les algorithmes à décision centralisée, puis aborderons les approches semi-centralisées, avant de détailler les méthodes décentralisées, en distinguant successivement les variantes coopératives et non coopératives.

2.3.2.1 Centralisation

Les algorithmes centralisés regroupent l'ensemble des stratégies dans lesquelles la prise de décision est assurée par une entité unique. Cette entité centrale, souvent qualifiée de contrôleur ou répartiteur, interagit directement avec l'ensemble des ressources afin d'optimiser la distribution des tâches en fonction de l'état global du système.

Dans ce contexte, nous nous intéressons aux algorithmes centralisés capables d'adapter leurs décisions en temps réel face aux variations de charge des ressources et à l'évolution du flux de travail entrant. Parmi ces algorithmes, les stratégies basées sur des seuils constituent une famille emblématique. De manière générale, dans ce type d'approche, le répartiteur central s'appuie sur un seuil de charge ou de performance défini pour

chaque ressource. Lorsqu'une ressource atteint ou dépasse ce seuil, elle est considérée comme surchargée, et les nouvelles tâches sont alors orientées en conséquence vers d'autres ressources.

En complément des approches à seuil, d'autres stratégies exploitent l'état instantané des ressources pour orienter l'affectation des tâches. C'est notamment le cas des algorithmes fondés sur la politique de la moindre connexion, qui consistent à attribuer les nouvelles tâches à la ressource présentant le nombre minimal de connexions actives. D'autres encore prennent en compte des critères combinés tels que le temps de communication, le temps de traitement ou le coût global d'exécution afin de minimiser le délai ou maximiser l'efficacité globale.

Dans cette section, nous analysons trois approches représentatives. Tout d'abord, une stratégie basée sur des seuils adaptatifs capable d'ajuster dynamiquement ses paramètres en fonction des fluctuations du système. Ensuite, un algorithme qui combine mesures heuristiques locales et évaluation du coût d'équilibrage, pour décider de manière informée des transferts de charge. Enfin, un module de décision central qui vient s'ajouter à une stratégie d'équilibrage simple (par exemple, un tourniquet), afin d'optimiser simultanément la répartition de la charge et l'allocation dynamique des ressources physiques telles que le processeur et la mémoire.

Seuil auto-apprenant L'approche proposée par GOLDSZTAJN et al. (2022) introduit un mécanisme d'équilibrage de charge reposant sur un seuil dynamique auto-apprenant. L'objectif principal consiste à optimiser la qualité de service perçue par l'utilisateur en assurant une répartition efficiente des tâches entre plusieurs groupes de serveurs.

Le principe de cette méthode repose sur l'utilisation d'un seuil ℓ, représentant un niveau de charge à partir duquel les décisions d'allocation sont orientées. Lorsqu'une tâche arrive dans le système, sa destination est déterminée selon l'état de charge relatif des groupes, selon trois cas distincts :

— Si au moins un groupe présente une charge strictement inférieure à ℓ, la tâche est assignée aléatoirement à l'un de ces groupes.

— Si tous les groupes possèdent au moins ℓ tâches, mais que certains en ont exactement ℓ, le choix est effectué aléatoirement parmi ces derniers.

— Si l'ensemble des groupes ont une charge strictement supérieure à ℓ, la sélection s'effectue de manière aléatoire parmi l'ensemble des groupes.

Afin de mettre en œuvre ce processus tout en minimisant les coûts de communication et de stockage, les auteurs proposent une implémentation fondée sur un système de jetons. Chaque groupe peut émettre au plus deux jetons distincts à destination du répartiteur central : un jeton vert, indiquant une charge strictement inférieure à ℓ, et un jeton jaune, signalant une charge inférieure à $\ell + 1$. Lorsqu'une tâche est soumise au répartiteur, celui-ci tente en priorité de l'acheminer vers un groupe disposant d'un jeton vert. À défaut, il sélectionne un groupe muni d'un jeton jaune. Si aucun jeton n'est disponible, le choix s'effectue alors de manière uniforme au hasard.

Cette architecture présente l'intérêt notable de réduire la quantité d'informations à échanger et de simplifier la prise de décision du répartiteur, tout en maintenant une répartition efficace de la charge.

Néanmoins, une limite inhérente à cette approche réside dans le fait que le seuil ℓ constitue un paramètre

sensible à la charge entrante, laquelle peut être inconnue ou sujette à de fortes variations dans des environnements dynamiques. Pour remédier à ce problème, les auteurs introduisent une règle de contrôle adaptative permettant de faire évoluer le seuil en temps réel, à l'arrivée de chaque tâche. Cette évolution repose sur une analyse de l'état global du système, en particulier le nombre total de groupes n et leur niveau de charge respectif. Trois scénarios sont envisagés :

— Le seuil est incrémenté de 1 lorsque la quasi-totalité des groupes présentent une charge strictement supérieure à $\ell + 1$.

— Le seuil est décrémenté de 1 lorsque la proportion de groupes sous-chargés (ayant strictement moins de ℓ tâches) est inférieure à un seuil $\alpha \in [0; 1]$, défini par au préalable.

— Dans les autres situations, le seuil demeure inchangé.

Ce mécanisme d'ajustement permet au système de s'adapter aux variations temporelles de la charge ainsi qu'aux modifications du nombre de groupes de serveurs disponibles, garantissant un équilibrage performant face aux dynamiques imprévisibles d'un environnement incertain.

Équilibrage de charge dynamique global centralisé ZAKI et al. (1996) proposent une stratégie d'équilibrage de charge adaptée à divers niveaux de centralisation, déclinée en quatre variantes. Dans cette section, nous nous concentrons sur la première approche : l'équilibrage de charge dynamique global centralisé.

Dans ce modèle, le contrôleur central, qui est également une ressource de traitement, interagit directement avec l'ensemble des ressources du système. La collecte des informations nécessaires à la prise de décision repose sur un mécanisme de synchronisation déclenchée par la première ressource ayant terminé une tâche. Lors de cette synchronisation, chaque ressource interrompt momentanément son activité et transmet au contrôleur un profil de performance correspondant à son taux de traitement (mesuré en nombre d'instructions par seconde) depuis la dernière synchronisation.

Les auteurs modélisent le coût de synchronisation comme la somme du coût d'une communication un-vers-tous (de la ressource initiatrice vers toutes les autres) et d'une communication tous-vers-un (des ressources vers le contrôleur central). Une fois les informations collectées, le contrôleur calcule une nouvelle répartition des tâches en se basant sur les performances passées pour estimer les performances futures des ressources.

Avant de procéder à la migration des tâches, le contrôleur mène une analyse de rentabilité visant à évaluer si les gains attendus en termes de performances justifient les coûts associés aux déplacements de tâches. Ce coût dépend de la quantité de travail à transférer, du nombre de messages échangés et des caractéristiques du réseau (telles que la latence et la bande passante). Ce mécanisme vise à éviter des réallocations inefficaces, telles que le déplacement de petites charges sans bénéfice notable ou, à l'inverse, une réorganisation massive pour un gain marginal.

Le déplacement effectif des tâches n'est déclenché que si la rentabilité estimée dépasse un seuil prédéfini. Lorsque cette condition est satisfaite, le contrôleur communique à chaque ressource les instructions nécessaires, spécifiant les tâches à transférer et les destinations correspondantes. Les ressources émettrices envoient alors directement les données aux ressources réceptrices, qui reprennent leur activité une fois l'ensemble des données attendues reçues.

Bien que cette stratégie permette d'atteindre une répartition de charge proche de l'optimum, les auteurs soulignent qu'elle peut entraîner un coût de communication et de synchronisation particulièrement élevé par rapport à ses variantes plus distribuées. Afin de réduire ces coûts, une variante reposant sur une prise de décision locale est proposée.

Dans cette approche, le système est subdivisé en sous-groupes de ressources, chacun fonctionnant comme un système d'équilibrage dynamique autonome. Les ressources appartenant à un même groupe peuvent échanger leurs états et transférer des tâches entre elles, mais aucune communication inter-groupes n'est autorisée. Chaque groupe est doté de son propre contrôleur central, chargé d'orchestrer localement l'équilibrage de la charge.

Cette organisation permet de réduire les coûts de communication et de synchronisation. Cependant, elle présente également certaines limites : l'équilibrage global du système peut s'en trouver affecté. En effet, l'absence de coordination entre les groupes peut conduire à des déséquilibres où certains groupes demeurent surchargés tandis que d'autres disposent de ressources sous-utilisées. De plus, la convergence vers une solution équilibrée peut s'avérer plus lente en comparaison avec la stratégie entièrement centralisée.

Enfin, les auteurs proposent également des variantes entièrement décentralisées de l'algorithme, dont les principes et le fonctionnement seront examinés dans la section dédiée à la décentralisation.

Module de décision central de répartition de charge Le module de décision central de répartition de charge (Central Load Balancing Decision Module), tel que décrit par RADOJEVIĆ et ŽAGAR (2011), ne constitue pas à proprement parler un algorithme d'équilibrage de charge. Il s'agit plutôt d'un composant externe qui vient s'ajouter à une infrastructure centralisée préexistante afin d'optimiser dynamiquement l'équilibrage de charge en réponse à l'état réel du système.

Ce module s'inscrit néanmoins pleinement dans le paradigme de la répartition centralisée, dans la mesure où il agit en tant que contrôleur centralisé et que ses décisions influencent directement la distribution des requêtes à travers le système. Sa particularité réside dans sa capacité à interagir avec l'ensemble des composants du système pour collecter des données, analyser leur état, et orchestrer des ajustements de manière proactive.

Afin de remplir sa fonction, le module surveille divers indicateurs pertinents, tels que le trafic au niveau des équilibreurs ainsi que l'utilisation des ressources matérielles, incluant serveurs physiques, processeurs, mémoire vive, etc. À partir des données collectées, il procède à des calculs internes, destinés à évaluer les déséquilibres potentiels ou réels. Il peut ensuite influencer les décisions des équilibreurs en leur transmettant des directives spécifiques.

Par exemple, dans le cas d'un algorithme de type tourniquet, le module peut demander à exclure temporairement une ressource surchargée de la rotation, jusqu'à ce qu'elle retrouve un état normal. Il surveille également l'expérience utilisateur, mesurée principalement par le temps de réponse global d'une requête (comprenant l'envoi de la tâche, son traitement, et le retour du résultat à l'utilisateur). Une augmentation significative de ce temps de réponse constitue un indicateur de dysfonctionnement, potentiellement lié à une surcharge ou à une latence réseau. En réaction, le module peut réorienter dynamiquement la charge vers d'autres ressources plus disponibles afin de restaurer la qualité de service.

La Figure 2.10 illustre le fonctionnement général d'un système intégrant un module de décision central.

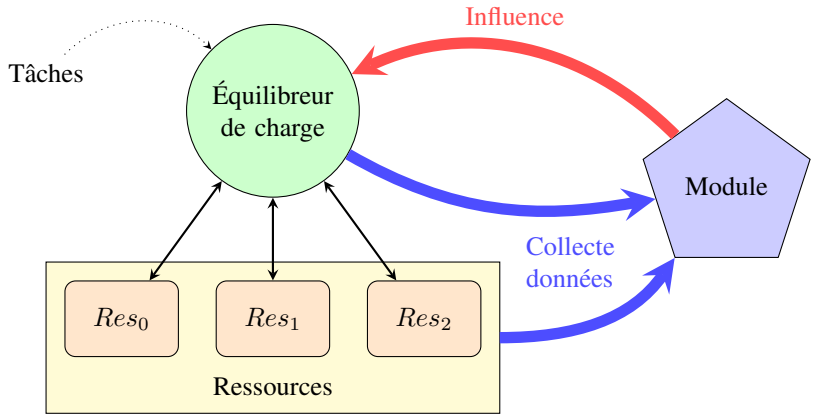


FIGURE 2.10 – Schématisation d'un système doté d'un module de décision central d'équilibrage de charge.

Bien que l'action du module s'exerce en premier lieu sur les équilibreurs de charge, son champ d'intervention s'étend également aux ressources elles-mêmes, que l'on désignera ici par le terme de serveurs virtuels. Un serveur virtuel représente une instance logicielle capable de traiter des requêtes, et s'exécute généralement sur un serveur physique, lequel peut héberger plusieurs instances virtualisées identiques. Ce modèle est conforme à l'architecture web semi-centralisée décrite en Section 2.1.2.2.

Grâce à la virtualisation, le module peut migrer dynamiquement des instances de serveurs virtuels d'un serveur physique à un autre. Par ailleurs, il est en mesure de réallouer les ressources matérielles (processeur, mémoire, etc.) entre les différentes instances selon les besoins observés.

La Figure 2.11 présente un exemple de migration. Le système comprend un module de décision centralisé, un équilibreur de charge, ainsi que deux serveurs physiques, notés SP_A et SP_B. Chacun héberge deux serveurs virtuels : SV_{A1}, SV_{A2}, SV_{B1} et SV_{B2}. Lorsque SV_{A1} subit une dégradation de performance, le module décide de migrer SV_{A2} vers SP_B, dont les ressources sont suffisantes. Il procède ensuite à une réallocation des ressources de SP_A au bénéfice exclusif de SV_{A1}, tandis que les ressources de SP_B sont réparties entre trois instances : SV_{B1}, SV_{B2} et SV_{A2} (désormais migré).

2.3.2.2 Semi-centralisation

La semi-centralisation désigne les stratégies d'équilibrage de charge combinant des éléments de prise de décision centralisée et décentralisée, dans l'objectif de tirer parti des avantages respectifs de ces deux approches. Ce type de paradigme repose le plus souvent sur une organisation hiérarchique de la structure décisionnelle : le système est alors composé de plusieurs contrôleurs centraux, chacun responsable d'un sous-groupe de ressources ou de contrôleurs, formant une arborescence logique jusqu'aux unités de calcul. Dans la majorité des cas, cette hiérarchie est préétablie. Toutefois, certains travaux, comme celui de SIOUTAS et al. (2022) avec le D³-Tree, proposent des mécanismes de maintien et d'adaptation dynamique de cette structure.

Les objectifs des stratégies hiérarchiques sont variés. Certaines cherchent à équilibrer la charge de travail entre les ressources, tandis que d'autres visent à répartir équitablement les ressources entre différentes branches

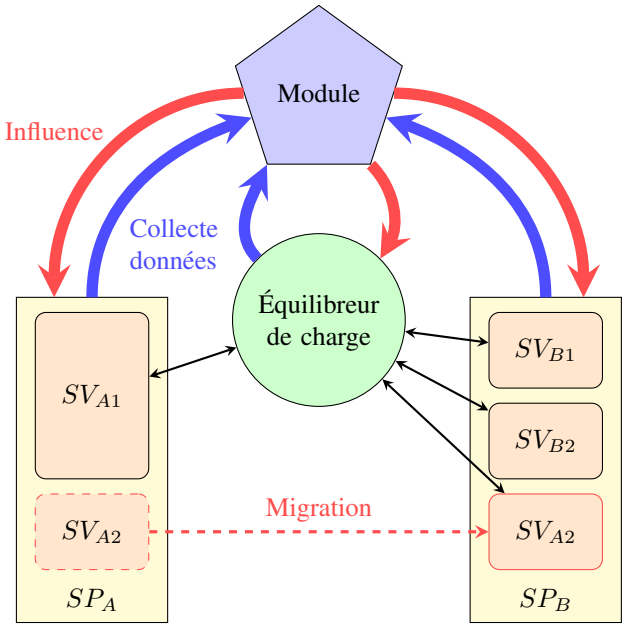


FIGURE 2.11 – Exemple de migration du serveur virtuel A2 depuis le serveur physique A vers le B par le module de décision central d'équilibrage de charge en réponse à une latence de A1.

de l'arbre pour faciliter la distribution. Enfin, certaines approches combinent ces deux objectifs. À ce titre, l'étude menée par PRIYA et GNANASEKARAN (2017) propose une classification de solutions couvrant ces différentes configurations, avec, entre autres, des structures à deux (MEGHARAJ & MOHAN, 2013) ou trois niveaux (S.-C. WANG et al., 2010).

Dans cette section, nous nous concentrerons particulièrement sur le D³-Tree, une solution capable de maintenir dynamiquement un arbre binaire parfait servant de surcouche de routage des requêtes vers les ressources situées en feuilles. Nous aborderons ensuite une stratégie d'équilibrage paresseuse, qui ne déclenche la redistribution des tâches que lorsqu'un déséquilibre significatif est détecté à une échelle locale ou globale.

Arbre dynamique, distribué et déterministe L'arbre dynamique, distribué et déterministe (Dynamic Distributed Deterministic Tree, D³-Tree) (SIOUTAS et al., 2022) est une structure de données distribuée qui agit comme une surcouche de contrôle au réseau de communication d'un système décentralisé. Son objectif est de structurer et restreindre les communications entre ressources afin d'optimiser les transmissions d'information. Ainsi, même si le réseau physique sous-jacent est pleinement connecté, la surcouche limite les communications aux liens jugés pertinents.

Dans le cadre des algorithmes étudiés, la surcouche prend la forme d'un arbre binaire parfait, où chaque nœud dispose exactement de deux fils et toutes les feuilles se trouvent à une même profondeur, comme illustré à la Figure 2.12 pour un arbre de hauteur 3. Le D³-Tree fournit une méthode distribuée permettant de maintenir dynamiquement cette structure d'arbre parfait au sein d'un réseau décentralisé. Celui-ci sert de canal de routage des tâches vers les feuilles, lesquelles, à leur tour, répartissent les tâches parmi les ressources qu'elles regroupent. La structure est hiérarchique, dynamique et auto-gérée, avec des algorithmes déterministes garantissant une complexité maximale de $\mathcal{O}(log(N))$ pour toute opération, où N est le nombre total de nœuds.

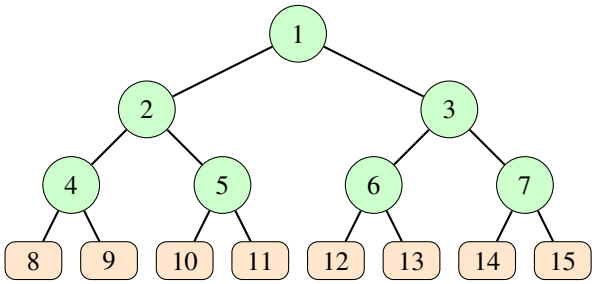


FIGURE 2.12 – Exemple d'arbre binaire parfait de hauteur 3.

Pour appréhender le fonctionnement du D³-Tree, il est nécessaire de présenter au préalable son prédécesseur, le D²-Tree (Deterministic Decentralized Tree) (BRODAL et al., 2015), dont il constitue une extension.

Le D²-Tree a été conçu pour l'organisation et la recherche de données dans des réseaux de type pair-à-pair. Il repose sur une architecture à deux niveaux. Le niveau supérieur est un arbre binaire parfait de taille $log(N)$, destiné à accélérer les recherches. Le niveau inférieur se compose de conteneurs ("buckets"), c'est-à-dire d'ensembles de $log(N)$ nœuds regroupés autour d'une feuille représentante dans l'arbre supérieur. Les conteneurs sont des listes doublement chaînées, leur premier élément étant connecté directement à la feuille correspondante.

Chaque nœud de l'arbre maintient plusieurs types de connexions :

— Vers son père et ses enfants ;

— Vers ses voisins dans un parcours infixe (gauche, racine, droite) ;

— Vers la feuille la plus à gauche et la plus à droite de son sous-arbre ;

— Vers des nœuds du même niveau avec des sauts exponentiels (2^n-ièmes voisins à gauche et à droite).

Ces connexions respectent des propriétés strictes : par exemple, si deux nœuds sont connectés, leurs pères le sont également, et de même pour leurs frères gauches et droits.

La plage de données gérée par un nœud correspond à l'union des plages de ses descendants. La navigation dans l'arbre se fait donc efficacement par subdivisions successives (hiérarchie).

L'arbre supporte diverses opérations décentralisées de maintien : lorsqu'un nouveau nœud rejoint le réseau, il est intégré à un conteneur existant via une redirection effectuée par un nœud arbitraire. Lorsqu'un nœud est supprimé, il est remplacé par l'un de ses voisins infixes. Les tables de routage sont mises à jour dynamiquement pour conserver la cohérence structurelle.

Le D²-Tree assure également un équilibrage de la charge grâce à des mécanismes de pondération. Chaque nœud calcule un poids correspondant à la charge que doit gérer son sous-arbre, et un rééquilibrage est déclenché si l'écart de densité (ratio entre le poids d'un nœud et la taille cumulée des conteneurs de son sous-arbre) entre deux frères dépasse un certain seuil. Par ailleurs, un système d'extension et de contraction ajuste la hauteur de l'arbre en fonction de la taille des conteneurs.

Ainsi, cette structure parvient à combiner centralisation de la recherche et décentralisation du maintien, assurant une répartition équilibrée de la charge tout en tolérant une certaine dynamique dans le réseau. Cependant, le D²-Tree présente une fragilité importante face aux défaillances, notamment si une feuille tombe en

panne : le conteneur qu'elle représente devient alors inatteignable. C'est précisément cette faiblesse que vient corriger le D³-Tree.

Dans le D³-Tree, chaque feuille étend ses connexions aux conteneurs des feuilles référencées dans sa table de routage (ses 2^n-ièmes voisins). Ainsi, un conteneur reste accessible même si sa feuille représentante est défaillante, et l'arbre peut être réorganisé pour exclure dynamiquement les nœuds défaillants.

SIOUTAS et al. (2022) démontrent que cette amélioration renforce considérablement la robustesse du système sans pour autant dégrader ses performances par rapport à la structure initiale.

Équilibrage de charge périodique hiérarchique L'équilibrage de charge périodique hiérarchique (Periodic Hierarchical Load Balancing) proposé par ZHENG et al. (2011) a été conçu pour améliorer les performances des supercalculateurs de grande échelle. Cette approche vise à combiner les avantages des stratégies centralisées et distribuées, tout en limitant leurs inconvénients : la faible scalabilité des premières et les coûts computationnels élevés des secondes.

Le système repose sur une division des ressources en groupes autonomes, eux-mêmes organisés en une hiérarchie arborescente. Chaque groupe fonctionne de manière indépendante selon un schéma centralisé localement, tout en s'insérant dans une structure globale décentralisée. Ainsi, chaque nœud de l'arbre, qualifié de chef de groupe (group leader) par les auteurs, est responsable de l'équilibrage de charge dans son propre sous-arbre, indépendamment des autres branches. Comme évoqué en Section 2.1.2.2, cette approche permet de réduire les ressources mémoire et le temps d'exécution requis pour réaliser l'équilibrage, chaque chef de groupe ne gérant qu'un sous-ensemble limité de ressources.

Le terme "périodique" renvoie au fait que l'équilibrage n'est pas continu, mais effectué ponctuellement, lorsque cela est jugé nécessaire. Un rééquilibrage est déclenché si un déséquilibre de charge est détecté au sein d'un sous-arbre. La détection repose sur une communication ascendante de la charge entre les nœuds : chaque chef de groupe agrège les informations provenant de ses enfants pour évaluer l'état global de son sous-arbre. Un rééquilibrage est initié lorsque la charge d'un enfant diverge significativement de la moyenne du groupe, à condition que le gain de performances attendu soit supérieur au coût de l'opération (collecte des données, exécution de l'algorithme, migration des tâches).

Pour estimer ce gain, l'algorithme s'appuie sur une forme de persistance des tâches : bien que leur durée ou leur complexité exacte soit inconnue, les tâches ont tendance à avoir une structure ou un comportement similaire au cours du temps. Le rééquilibrage s'appuie alors sur un mécanisme de jetons contenant les métadonnées des tâches, qui sont redistribués de manière centralisée par chaque chef de groupe. Les tâches elles-mêmes ne sont déplacées qu'une fois la redistribution des jetons complétée, ce qui permet de limiter le coût de communication.

Grâce à cette architecture hiérarchique, l'approche permet de réaliser des ajustements localement, tout en conservant la capacité à réagir globalement à des déséquilibres plus étendus. Elle illustre ainsi un compromis efficace entre centralisation et décentralisation, dans le but d'assurer un équilibrage de charge scalable, flexible et performant.

2.3.2.3 Décentralisation coopérative

La décentralisation coopérative regroupe les stratégies déployées dans des environnements distribués où l'ensemble des entités du système collaborent activement pour assurer l'équilibrage de charge. Ce mode de fonctionnement repose sur le partage d'informations entre les ressources, telles que leur niveau de charge, leur capacité de traitement, ou encore les métadonnées associées aux tâches.

En fonction du mécanisme de diffusion adopté, les ressources peuvent disposer d'informations locales, issues de leur voisinage immédiat, ou d'informations globales, lorsque la propagation de l'information est conçue pour couvrir l'ensemble du système. Sur la base de ces données, chaque ressource est en mesure de prendre des décisions d'équilibrage de manière autonome, sans supervision centralisée.

La communication entre ressources peut s'opérer selon plusieurs modalités :

— par communication indirecte ou environnementale, où les entités modifient leur environnement commun (par exemple via des dépôts de traces ou de signaux) pour influencer les décisions futures ;
— par diffusion structurée de l'information, à l'aide de protocoles de communication dédiés ;
— ou encore par partage direct entre pairs, sous forme d'échanges explicites de données.

Ces mécanismes de partage peuvent aboutir soit à un consensus global, coordonnant l'ensemble des décisions, soit à des prises de décisions locales, potentiellement indépendantes mais cohérentes grâce à la coopération.

Dans la suite, nous nous intéresserons à trois algorithmes représentatifs de cette approche : l'optimisation par colonies de fourmis et le comportement coopératif des abeilles appliqués à l'équilibrage de charge en temps réel par communication environnementale, ainsi qu'une version distribuée de l'algorithme d'équilibrage dynamique centralisé présenté précédemment, basé sur l'obtention d'un consensus global par communication directe.

Optimisation par colonies de fourmis en temps réel L'optimisation par colonies de fourmis, que nous avons étudiée précédemment, est une méthode d'optimisation au sens strict. Cependant, elle a été adaptée à des fins d'équilibrage de charge décentralisé en temps réel (KATYAL & MISHRA, 2013 ; LI et al., 2011 ; LIU et al., 2006).

Le problème à résoudre est ici l'équilibrage des tâches sur les ressources d'un système décentralisé. Ce système constitue l'espace exploratoire des fourmis : chaque ressource représente une solution potentielle pour le dépôt d'une tâche. Les fourmis servent alors de "véhicule" aux tâches, qu'elles déplacent de manière stochastique dans le système. Le déplacement est influencé non seulement par les phéromones déposées, mais également par des informations heuristiques locales, telles que la capacité de la ressource, les tâches en attente ou l'état de charge des ressources voisines.

Les fourmis tendent à se diriger vers les ressources les plus susceptibles de traiter rapidement leur tâche, c'est-à-dire celles faiblement chargées ou plus performantes. L'exploration d'une fourmi prend fin dès qu'une condition liée à la ressource courante est remplie, par exemple lorsque sa charge est inférieure à celle des ressources voisines. Elle affecte alors sa tâche à cette ressource, puis dépose des phéromones sur le chemin emprunté. Comme dans l'algorithme d'origine, ces phéromones s'évaporent progressivement. Une fois leur

tâche délivrée, les fourmis disparaissent.

Cette communication indirecte via les phéromones permet aux tâches d'atteindre dynamiquement la ressource la plus adaptée, et produit une auto-organisation coopérative du système. L'évaporation des phéromones et l'utilisation d'informations heuristiques locales assurent à l'algorithme une capacité d'adaptation aux variations de charge dans le système.

Une autre approche, complémentaire, vise à optimiser les communications entre ressources dans un environnement distribué dynamique, afin d'améliorer la migration des tâches. Les travaux de CARDON et al. (2006) et DUTOT (2005) proposent un algorithme bio-inspiré appelé AntCO², qui combine optimisation par colonies de fourmis et mécanismes de compétition. L'objectif est de concilier minimisation des communications et équilibrage de charge, sans recourir à une fonction objectif explicite. L'organisation émerge alors de manière autonome.

Le système est modélisé sous forme de graphe dynamique, où les sommets représentent les ressources et les arêtes les communications. Chaque sommet constitue une colonie identifiée par une couleur. Des fourmis partent de leur colonie et déposent des phéromones colorées le long de leur chemin. La quantité de phéromones est déterminée dynamiquement et celles-ci s'évaporent avec le temps. La couleur dominante des phéromones présentes sur les arêtes d'un sommet définit sa propre couleur.

Lorsqu'une fourmi se déplace, elle privilégie les arêtes où la concentration de phéromones de sa propre couleur est la plus élevée. Si cette concentration est trop faible, la fourmi meurt et une nouvelle naît ailleurs dans le graphe (mécanisme de "mort et éclosion"). La compétition entre colonies conduit à une agrégation des ressources communiquant intensivement, qui tendent à adopter la même couleur. Cette structuration dynamique du graphe favorise la migration optimale des tâches tout en réduisant les communications.

L'algorithme des abeilles en temps réel L'algorithme des abeilles, présenté auparavant comme une méthode d'optimisation stricte, a également été adapté à l'équilibrage de charge décentralisé en temps réel. La version adaptée, nommée butinage des abeilles (Honeybee Foraging) (KATYAL & MISHRA, 2013 ; RANDLES et al., 2010 ; SESUM-CAVIC & KÜHN, 2010a, 2010b), modélise chaque ressource (nœud) du système sous forme de fleur (file d'attente des tâches) et de ruche (système de calcul local). Les fleurs contiennent du nectar (les tâches), que les abeilles transfèrent afin d'équilibrer la répartition entre les ruches, qui les traiteront ensuite.

Chaque ruche dispose de trois types d'abeilles : les butineuses (ou exploratrices), les suiveuses et les receveuses. Ces dernières modélisent le traitement local du nectar. Les butineuses explorent le système pour identifier une ruche partenaire avec laquelle échanger du nectar. Une fois la cible identifiée, les suiveuses réalisent le transfert de tâches entre les ruches.

Le déclenchement de la recherche d'une ruche partenaire dépend d'une politique libre. Il peut s'agir, par exemple, d'un déclenchement par les nœuds surchargés, sous-chargés, ou encore en prévention par ceux proches de la surcharge. En fonction de cette politique, les cibles seront choisies parmi les nœuds présentant un état opposé à celui du nœud initiateur.

Lors de leur exploration, les butineuses choisissent leur direction à l'aide d'une règle de transition d'état P_{ij}, où i et j désignent respectivement leur position actuelle et une destination possible. Cette règle, intro-

duite par WONG et al. (2008), s'inspire du comportement naturel des abeilles qui effectuent une "danse" pour indiquer les zones les plus riches en nectar (BIESMEIJER & SEELEY, 2005). Une butineuse ayant trouvé un partenaire pertinent retourne à sa ruche et en partage les caractéristiques : qualité du partenaire, chemin emprunté, distance, état des nœuds visités.

À chaque visite, la butineuse évalue la convenance du nœud rencontré via une fonction $\delta(x)$, où x est une mesure de l'état du nœud. Selon SESUM-CAVIC et KÜHN (2010b), cette évaluation peut être donnée par $x = \frac{cp/p}{1-ch}$, où cp est la complexité des tâches, p la puissance de calcul, et ch la charge du nœud. Plus x est proche de 1, plus le partenaire est considéré comme adéquat.

La butineuse calcule ensuite une valeur de fitness $f(x) = \frac{\delta(x)}{distance}$, représentant le rapport entre la qualité du partenaire et le coût d'accès. Si cette valeur est élevée par rapport à la fitness moyenne de la ruche (f_{ruche}), la butineuse recrute des suiveuses pour transférer des tâches. Elle influence également les explorations futures. En revanche, si sa fitness est faible, elle peut devenir suiveuse elle-même.

Ce mécanisme d'exploration, d'évaluation et de recrutement induit une auto-organisation efficace du système. Il permet une répartition dynamique des charges, adaptée aux capacités et aux besoins des nœuds à tout moment.

Équilibrage de charge dynamique global distribué Nous nous intéressons à présent à la version décentralisée de l'équilibrage de charge dynamique introduite par ZAKI et al. (1996), précédemment présentée dans la section consacrée aux algorithmes centralisés. Le principe fondamental de l'algorithme demeure identique, à l'exception notable que le contrôleur central est ici répliqué sur l'ensemble des ressources.

Lors du déclenchement d'une synchronisation, qui survient dès qu'une ressource termine une tâche, chaque ressource diffuse son profil de performance à l'ensemble des autres. Le coût de synchronisation se compose alors d'une communication un-à-tous (déclenchement de la synchronisation) suivie d'une communication tous-à-tous (diffusion des profils de performance entre toutes les ressources). Chaque ressource, disposant de la vue complète de l'état du système, peut ainsi calculer localement une nouvelle distribution des tâches.

Si le bénéfice escompté de la redistribution des tâches excède le coût de communication associé au transfert des données, alors l'équilibrage est effectivement déclenché. Chaque ressource détermine son rôle à partir de la nouvelle distribution : une ressource identifiée comme émettrice (c'est-à-dire disposant d'un excédent de charge) transfère directement ses tâches excédentaires à une ressource réceptrice (dont la charge cible est supérieure à sa charge actuelle).

Cette approche repose donc sur un partage globalisé de l'information et une coopération entre les ressources, impliquant une interconnexion complète du réseau. Un tel modèle entraîne un coût communicationnel potentiellement élevé, notamment dans les systèmes de grande taille. Cependant, les auteurs soulignent que cette solution offre, dans la plupart des cas, les meilleures performances en termes d'équilibrage et de temps de réponse, comparée à la version centralisée.

À l'instar de la stratégie centralisée, les auteurs proposent également une seconde variante décentralisée. Celle-ci reprend le même mécanisme, mais appliqué au sein de groupes de ressources fermés. Ainsi, chaque groupe procède à un équilibrage interne sans coordination avec les autres. Bien que cette méthode permette

de réduire les coûts de communication, elle reste vulnérable aux déséquilibres globaux induits par l'absence d'échanges inter-groupes, de la même manière que pour sa version centralisée en groupes.

2.3.2.4 Décentralisation non-coopérative

Les algorithmes décentralisés non coopératifs forment une classe de stratégies dans laquelle aucune coordination explicite n'est établie entre les ressources du système. Les communications, lorsqu'elles existent, sont minimales ou strictement locales, ce qui rend les décisions d'équilibrage entièrement autonomes et indépendantes. Dans la majorité des cas, chaque ressource prend ses décisions en se basant exclusivement sur des informations locales : son niveau de charge, sa file d'attente, ou son état d'inactivité.

Cette approche se retrouve notamment dans plusieurs mécanismes :

— les mécanismes de vol de travail (work stealing), dans lesquels une ressource inactive initie elle-même le transfert en récupérant du travail depuis une autre ;

— les mécanismes de partage de travail (work sharing), qui au contraire délèguent automatiquement les nouvelles tâches vers d'autres ressources dès leur arrivée, sans attendre une demande ou une concertation ;

— les algorithmes à seuil, où les tâches en excès sont transférées vers d'autres ressources, soit de manière aléatoire, soit vers des nœuds dédiés appelés puits (sinks), comme nous le verrons plus loin.

Dans ces systèmes, la sélection de la ressource cible repose fréquemment sur un sondage aléatoire du voisinage, sans garantie de charge optimale, mais avec l'objectif de réactivité et de simplicité.

Dans la suite, nous étudierons trois algorithmes représentatifs de cette approche : la stratégie d'équilibrage par vol de travail, un échantillonnage aléatoire biaisé, proposant un mécanisme de partage de tâches intelligent, et enfin une politique de déchargement distribuée fondée sur un seuil local dynamique.

Équilibrage par vol de travail Le vol de travail (work stealing) (BLUMOFE & LEISERSON, 1999) est un algorithme décentralisé initialement introduit pour le calcul multi-threadé dans le cadre de la parallélisation intra-processeur, c'est-à-dire entre les différents cœurs d'un même processeur. Le principe est simple : lorsqu'un cœur de processeur a épuisé sa pile de threads (tâches ou sous-tâches parallélisables), il tente de voler du travail à un autre cœur encore actif.

Pour ce faire, chaque cœur maintient une structure de données appelée file prête (ready deque), qui fonctionne comme une file à double extrémité : les nouveaux threads sont ajoutés par le bas, un cœur retire les threads à traiter également par le bas, tandis que les autres cœurs peuvent voler un thread par le haut de cette file.

Le traitement des threads suit trois comportements principaux :

— **Création de sous-thread :** le thread courant est replacé en bas de la file et le nouveau thread est exécuté immédiatement.

— **Fin du thread :** le cœur récupère le thread suivant dans sa file, ou initie un vol si elle est vide.

— **Blocage du thread :** il est temporairement mis de côté, appliquant le comportement de fin de thread, et réinséré en bas de la file une fois débloqué.

Lorsqu'un cœur devient inactif, il tente de voler un thread à un autre cœur, choisi uniformément au hasard. S'il trouve une victime ayant une file non vide, il prend le thread le plus en haut. Sinon, il sélectionne un autre cœur et recommence. Ce mécanisme implique une interconnexion complète entre les ressources du système.

Ce modèle présente de bonnes performances et réduit le nombre de migrations de tâches comparé à des stratégies comme le partage de travail (work sharing), où les threads sont immédiatement migrés dès leur création. Le vol de travail n'opère des transferts que lorsqu'ils sont strictement nécessaires, c'est-à-dire lorsqu'une ressource devient inactive.

Ce modèle a été étendu aux environnements distribués, notamment par K. WANG et al. (2014), pour s'adapter à des applications à fort volume de données. Dans ces contextes, le simple vol de tâches pose un nouveau défi : la localisation des données. Déplacer une tâche peut impliquer un transfert massif de données, ce qui peut annuler les bénéfices de l'équilibrage de charge.

Pour pallier ce problème, les auteurs introduisent un vol de travail intelligent. Chaque nœud dispose désormais de deux files : une liste locale, réservée aux tâches devant être exécutées localement en raison du poids de leurs données, et une liste partageable, contenant les tâches à faible empreinte de données, donc transférables à moindre coût.

Lorsqu'une tâche arrive sur un nœud, celui-ci estime le coût de transfert des données associées. Si ce coût est inférieur à un seuil t, la tâche est placée dans la liste partageable. Sinon, elle est soit traitée localement, soit redirigée vers un autre nœud mieux placé selon la localisation des données.

Le choix de la victime n'est plus purement aléatoire : un nœud inactif (voleur) parcourt un sous-ensemble aléatoire de nœuds et choisit celui ayant le plus de tâches partageables. Un mécanisme de temporisation adaptative est aussi introduit : plus un nœud échoue à voler, plus il attend avant de réessayer. En cas de succès, l'intervalle entre les tentatives est réinitialisé à une valeur faible.

Une politique de répartition flexible et sensible à la localité des données permet aux nœuds de rééquilibrer leurs files en cas de surcharge. Un nouveau seuil tt est défini comme temps maximal d'exécution acceptable pour la liste locale. Le temps d'exécution de la liste est estimé par rapport au débit (tâches par seconde) passé du nœud. Si l'estimation dépasse ce seuil, les tâches du bas de cette file sont déplacées vers la liste partageable, permettant leur redistribution.

Enfin, un système de stockage clé-valeur distribué est utilisé pour partager entre tous les nœuds les métadonnées des tâches, incluant leurs dépendances et l'emplacement de leurs données.

Échantillonnage aléatoire biaisé L'échantillonnage aléatoire biaisé (Biased Random Sampling) (KATYAL & MISHRA, 2013 ; RAHMEH et al., 2008 ; RANDLES et al., 2010) repose sur une marche aléatoire dans un graphe virtuel représentant les ressources du système. Il vise à équilibrer les charges tout en structurant les communications de manière décentralisée et dynamique.

Le système est modélisé comme un graphe orienté dont les sommets représentent les ressources. Ce graphe est initialisé aléatoirement. Le degré entrant d'un sommet reflète sa capacité initiale : plus une ressource est puissante, plus elle possède d'arêtes entrantes. Les arêtes sortantes sont déterminées indirectement, selon la construction des autres arêtes entrantes. Le degré entrant évolue au fil du temps et finit par refléter la disponi-

bilité actuelle de la ressource.

Chaque ressource n'a connaissance que de son voisinage immédiat (entrées et sorties), ce qui permet une gestion locale du graphe, par exemple via une table de routage.

Lorsqu'une tâche arrive, la ressource déclenche une marche aléatoire à travers le graphe. À chaque étape, un voisin est choisi aléatoirement via une arête sortante du sommet courant. La marche se limite à $w = log(n)$ sauts, où n est le nombre total de ressources. Une fois la marche terminée, la tâche est affectée à la ressource ayant la meilleure disponibilité (c'est-à-dire le degré entrant le plus élevé) parmi celles visitées. Ce choix constitue le biais par rapport à un échantillonnage aléatoire simple.

Pour refléter cette affectation, la ressource cible supprime une de ses arêtes entrantes, réduisant ainsi sa disponibilité future. Une fois la tâche traitée, elle recrée une arête entrante depuis la ressource qui avait initié la marche. Cela augmente sa disponibilité et sa probabilité d'être sélectionnée à nouveau.

Ce modèle d'adaptation du graphe, combiné au biais dans la sélection des ressources, permet une auto-organisation du système en fonction des charges. Il pourrait cependant être amélioré par l'introduction d'autres critères de sélection : capacité réelle, géographie du réseau (latence), ou en le couplant avec un mécanisme de clustering actif tel que proposé par SAFFRE et al. (2009), afin de regrouper les services similaires et optimiser encore les parcours aléatoires.

Politique de déchargement distribuée basée sur un seuil La politique de déchargement distribuée basée sur un seuil (Decentralized Threshold-based Offloading Policy), proposée par QIN et al. (2021, 2023a, 2023b), s'inscrit dans le contexte du cloud computing mobile. L'architecture du système repose sur une hiérarchie composée de dispositifs mobiles à capacité de calcul limitée, susceptibles de recevoir des charges de travail, et d'un ensemble de serveurs mutualisés disposant de ressources computationnelles substantielles.

Cette stratégie vise à décharger dynamiquement les tâches des appareils mobiles vers les serveurs, en particulier lorsque les premiers sont en situation de surcharge. Chaque appareil mobile prend de manière autonome la décision de déléguer ou non une tâche, en se fondant sur un seuil propre maintenu localement. Deux mécanismes de mise à jour de ce seuil sont proposés, tous deux validés à la fois théoriquement et empiriquement (via des simulations), comme étant efficaces en termes de performances globales du système.

La première méthode, dite de mise à jour distribuée du seuil (Distributed Threshold Update), repose sur une combinaison de données locales propres à chaque appareil et d'une estimation globale de l'état des serveurs. Les données locales incluent, entre autres, la longueur de la file d'attente, la latence liée au déchargement de tâches, ainsi que la consommation énergétique. À chaque itération, le groupe de serveurs calcule une estimation de son taux d'occupation global, fondée sur les seuils reçus des appareils, puis diffuse cette information à l'ensemble des dispositifs mobiles. Ces derniers l'intègrent, conjointement avec leurs propres données locales, dans un problème d'optimisation visant à déterminer le seuil de déchargement qui minimise leur coût opérationnel. Ce processus est itératif et se poursuit jusqu'à ce que la variation entre deux estimations successives de l'occupation des serveurs soit inférieure à une tolérance ϵ. L'objectif de cet algorithme est d'atteindre un équilibre distribué, dans lequel aucun appareil n'a intérêt à modifier unilatéralement son seuil.

La seconde approche, nommée mise à jour itérative du seuil (Iterative Threshold Update), suit un principe

analogue, mais introduit une méthode d'optimisation incrémentale. Elle consiste à ajuster progressivement le seuil de déchargement dans le but de minimiser une fonction de coût propre à chaque appareil. Cette fonction combine le temps de traitement local, le coût d'utilisation des serveurs cloud, et l'état estimé du système. L'occupation des serveurs est inférée à partir des seuils des appareils à travers une probabilité de déchargement, elle-même dérivée du taux d'arrivée des tâches et de la longueur moyenne des files d'attente. L'algorithme commence par une phase d'optimisation gourmande, visant une amélioration rapide, suivie d'ajustements incrémentaux (+1 ou -1 sur le seuil) réalisés à chaque itération. Ce processus se poursuit jusqu'à l'atteinte d'un critère d'arrêt prédéfini, tel qu'un nombre fixe d'itérations ou la stabilisation du seuil.

En définitive, cette politique établit un schéma de déchargement totalement décentralisé, où chaque appareil prend ses décisions localement, sans communication directe avec les autres dispositifs, ni coordination explicite entre eux. Le seul échange d'information transite via les estimations globales fournies par le groupe de serveurs (qui ne prend aucune décision), garantissant ainsi une scalabilité élevée du système et une réduction substantielle de la surcharge de communication.

2.4 L'auto-organisation pour de la répartition dynamique

L'auto-organisation est un phénomène très présent dans les systèmes naturels, et plus largement dans les systèmes complexes (BOOLCHAND et al., 2005 ; HEYLIGHEN, 2009a, 2009b ; ISAEVA, 2012 ; LEVIN, 2005). Ces systèmes sont souvent décrits comme se situant dans un état hors de l'équilibre (PRIGOGINE, 1978), potentiellement dans certains cas au bord du chaos, où une perturbation minime peut entraîner une transformation drastique de leur configuration globale. Ce sont des systèmes dissipatifs typiquement ouverts traversés par des flux d'énergie, de matière ou d'information qui échangent avec leur environnement.

L'auto-organisation se définit comme l'apparition de structures spontanées ou de comportements globaux cohérents à partir d'interactions locales sans intervention externe ni contrôle centralisé. Ce caractère spontané découle du fait qu'aucun agent (interne ou externe au système) ne dirige ou ne coordonne explicitement le processus : celui-ci émerge collectivement, via l'agrégation non linéaire de causes locales. Ce mécanisme permet l'émergence de d'états ou de structures complexes imprévues à partir de règles simples (STROGATZ, 2024). Un corollaire fondamental est la robustesse structurelle. Dans un système suffisamment vaste, tout agent peut être retiré ou remplacé sans compromettre la dynamique globale.

Il s'agit d'un processus collectif, massivement parallèle et distribué, dans lequel chaque agent du système contribue également à l'organisation résultante. Les interactions s'effectuent d'abord localement, entre agents voisins, tandis que les agents éloignés agissent de manière indépendante. Cependant, une modification locale peut se propager en cascade et affecter des zones éloignées, générant ainsi une cohérence à l'échelle globale. De plus, des perturbations aléatoires peuvent se produire, permettant au système d'explorer de nouvelles trajectoires .

La nature des interactions locales varie selon le domaine étudié. Par exemple, en physique, elles peuvent prendre la forme de diffusions thermodynamiques ; en biologie, par des mécanismes de sélection naturelle ou de rétroaction (positive ou négative) permettant une adaptation dynamique des agents. Ces interactions permettent

au système de générer de l'ordre à partir du désordre, qu'il soit d'origine locale ou globale.

Grâce à leur caractère distribué, les systèmes auto-organisés se révèlent hautement robustes face aux défaillances locales et aux perturbations extérieures. L'auto-organisation induit par ailleurs l'émergence d'un attracteur dans l'espace des états du système : une région stable vers laquelle convergent spontanément les dynamiques du système, et au sein de laquelle les agents tendent à se stabiliser. Cette propriété restreint cependant la liberté individuelle des agents à sortir de certains états une fois l'attracteur atteint.

On retrouve des dynamiques d'auto-organisation dans une grande diversité de disciplines. En biologie, elles s'observent dans la morphogenèse, l'organisation cellulaire, les réseaux neuronaux, les écosystèmes ou encore dans les mécanismes évolutifs. En physique et chimie, elles apparaissent dans des processus tels que la cristallisation, la magnétisation, ou les structures dissipatives. Dans les systèmes d'information, on les retrouve dans l'architecture de l'internet, les réseaux de connaissances ou encore les automates cellulaires. Même les systèmes économiques et sociaux, comme les marchés, les communautés de recherche (PERC, 2013) ou les civilisations, présentent des comportements auto-organisés.

En raison de leur robustesse et de leur capacité à s'adapter dynamiquement, les systèmes auto-organisés sont explorés depuis les années 1990 comme modèle pour l'équilibrage de charge dans des environnements distribués. WILLEBEEK-LEMAIR et REEVES (1993) proposent deux stratégies décentralisées reposant respectivement sur des modèles de diffusion (les tâches s'écoulent des nœuds surchargés vers les sous-chargés) et de gradient (les nœuds sous-chargés sollicitent du travail). JELASITY et al. (2004) démontrent l'efficacité des protocoles de bavardage pour répartir la charge de manière efficace. Des approches explicitement basées sur l'auto-organisation ont ensuite été proposées, comme celles de JIE HU (2006) et LAREDO et al. (2017), s'inspirant respectivement de la diffusion d'un fluide (stabilisation) et de la dynamique du tas de sable (éboulement des grains), pour modéliser la redistribution des tâches. Ces méthodes reposent sur des interactions locales dans le voisinage immédiat pour permettre au système global d'atteindre un équilibre de charge dynamique et distribué.

Plusieurs des algorithmes présentés dans les sections précédentes relèvent également de l'auto-organisation. C'est notamment le cas des méthodes bio-inspirées, telles que l'optimisation par colonies de fourmis ou le comportement de recherche de nourriture des abeilles, qui utilisent des signaux locaux (stigmergie ou communication dansée) pour coordonner la répartition des tâches en temps réel.

Même les approches dites non coopératives peuvent manifester une forme d'auto-organisation. Le vol de travail, par exemple, aboutit à une occupation continue des ressources par simple effet d'interaction locale. L'échantillonnage aléatoire biaisé, de son côté, permet un équilibrage spontané en acheminant les nouvelles tâches vers les ressources les moins chargées, sans coordination centrale.

2.5 Discussion des méthodes et des modèles

Dans ce chapitre, nous avons examiné les systèmes d'équilibrage de charge à travers les principaux paradigmes qui les structurent : l'environnement d'exécution, l'architecture de contrôle, et le mode de prise de

décision. Nous avons également présenté différentes métriques d'évaluation permettant de mesurer leur efficacité et leur pertinence selon le contexte.

Le choix d'une stratégie d'équilibrage doit être soigneusement réfléchi, en tenant compte à la fois (i) des caractéristiques spécifiques de l'environnement d'exécution et (ii) des objectifs de performance visés. Par exemple, dans un contexte stable et bien maîtrisé, une approche centralisée fondée sur des heuristiques simples s'avère souvent suffisante, car elle permet une prise de décision rapide avec un coût de communication réduit. En revanche, dans un environnement dynamique, instable ou imprévisible, il est préférable de privilégier des approches décentralisées capables d'adaptation face à l'évolution constante du système.

Les méthodes centralisées présentent généralement l'avantage d'un faible coût en communication et d'une décision rapide, mais au prix d'un coût computationnel élevé (du fait de la vue globale nécessaire) et d'une fragilité structurelle liée au point de défaillance unique qu'est le contrôleur central. À l'opposé, les méthodes décentralisées répartissent la charge de décision entre les ressources, souvent via des mécanismes d'interaction locale ou de coopération. Elles sont intrinsèquement plus robustes, mais peuvent nécessiter un temps de convergence plus long et un coût communicationnel accru.

Les approches hybrides, généralement fondées sur des architectures hiérarchiques, tentent de combiner les avantages des deux paradigmes. Elles permettent une prise de décision plus rapide que les méthodes purement décentralisées tout en offrant une meilleure robustesse que les approches centralisées. Toutefois, leur complexité de mise en œuvre est significative, notamment lorsqu'elles doivent s'adapter dynamiquement à l'évolution du système. Elles peuvent également entraîner une réduction des capacités de calcul disponibles, une fraction des ressources étant mobilisée pour la gestion de l'équilibrage lui-même.

Comme nous l'avons vu précédemment, les dynamiques d'auto-organisation offrent une alternative décentralisée efficace pour atteindre un état d'équilibre global par le biais d'interactions locales. Ce type de système, par sa robustesse intrinsèque et sa capacité d'adaptation, est particulièrement adapté aux environnements distribués et dynamiques. C'est cette approche que nous avons choisie d'explorer dans le cadre des présents travaux.

Dans les chapitres à venir, nous nous concentrerons sur le modèle du tas de sable introduit par Bak, Tang et Wiesenfeld (BAK et al., 1987), ainsi que sur ses principales évolutions. Ce modèle, fondé sur un automate cellulaire à règles locales simples, est capable de générer des comportements complexes et émergents, en lien direct avec les dynamiques d'auto-organisation.

Le tas de sable constitue une approche naturelle et dynamique de l'équilibrage de charge : les cellules de l'automate (modélisant les ressources de calcul) se déchargent spontanément sur leurs voisines dès qu'un seuil local est dépassé. Les grains de sable, assimilables à des tâches, se déplacent ainsi dans l'espace jusqu'à ce que le système atteigne un état de stabilité globale.

Ce modèle sera étudié en détail dans le chapitre suivant, ainsi que le phénomène dont il constitue l'une des manifestations emblématiques : la criticalité auto-organisée, étroitement liée aux propriétés adaptatives des systèmes complexes, et tout particulièrement ceux issus de la nature.

Chapitre 3

La criticalité auto-organisée

Table des matières du chapitre

Les systèmes complexes ont souvent des comportements émergents, issus d'interactions locales simples entre leurs composants. Ces dynamiques peuvent engendrer des structures globales, une adaptation spontanée ou encore des réponses qui semblent disproportionnées lors de perturbations minimes. Parmi les phénomènes caractéristiques de ces systèmes figure la notion de criticalité auto-organisée (Self-Organized Criticality, SOC) (BAK, 1996). Elle décrit la tendance de certains systèmes à évoluer naturellement vers un état critique, situé à la frontière entre ordre et chaos, où une perturbation mineure peut déclencher des réactions majeures affectant l'ensemble du système. Ce phénomène a des implications profondes dans divers domaines, allant des neurosciences à l'économie, aidant à expliquer des phénomènes tels que le fonctionnement cérébral (PLENZ et al., 2021), les krachs boursiers (BIONDO et al., 2015) et d'autres événements de grande envergure (MARKOVIĆ & GROS, 2014).

Cette dynamique d'auto-organisation peut s'avérer très intéressante dans le cadre de l'équilibrage de charge.

Un tel système disposant d'une caractéristique de ce type pourrait être capable de répartir dynamiquement la charge par des mécanismes intrinsèques, sans nécessiter une intervention externe. Ainsi, de nombreux systèmes naturels présentent les caractéristiques de la SOC. Ces dernières leur permettent d'être résilients et robustes en présence de perturbations. Elle pourrait être particulièrement utile et efficace pour les systèmes distribués, pouvant être sujets à des pannes, et ce malgré leurs topologies diverses. C'est pour cette raison que, dans ce chapitre, une attention toute particulière sera portée sur l'exploration du tas de sable (représentant majeur des modèles de SOC) dans diverses topologies au travers de plusieurs études.

Dans ce chapitre, le concept de SOC sera tout d'abord introduit. Nous nous intéresserons ensuite aux différents modèles de SOC, et plus particulièrement au modèle du tas de sable. Notre attention se portera ensuite sur les topologies des systèmes SOC et les impacts de ces dernières sur leur dynamique. Nous poursuivrons avec une analyse de la robustesse des systèmes SOC, en mettant en lumière comment leur capacité à s'auto-organiser contribue à leur résilience. Pour terminer, nous verrons comment le modèle du tas de sable, modèle de référence de la SOC, peut être adapté à l'équilibrage dynamique de charge, en exploitant ses propriétés intrinsèques pour optimiser la répartition des ressources dans des systèmes complexes.

3.1 Introduction à la criticalité auto-organisée

La criticalité auto-organisée, introduite en 1987 par Bak, Tang et Wiesenfeld (BAK et al., 1987), a été illustrée à travers un automate cellulaire connu sous le nom de modèle du tas de sable. Ce modèle s'inspire des dynamiques d'avalanches observées dans les milieux granulaires, et met en évidence comment des interactions locales simples peuvent conduire un système vers un état critique global. Le modèle du tas de sable simule des grains de sable déposés aléatoirement sur les cellules d'une grille. Cette grille est de dimension 2 et régulière. Elle définit un espace discret formé de cases carrées et un voisinage pour chacune des cases de la grille. Lorsque le nombre de grains sur un site dépasse un certain seuil, les grains basculent vers les sites voisins, provoquant potentiellement une réaction en chaîne ou une avalanche. Ce modèle simple mais puissant capture l'essence de la SOC et a été largement étudié pour comprendre la dynamique des états critiques dans une vaste gamme de disciplines.

En neurosciences, le concept de criticalité auto-organisée a contribué à la compréhension des avalanches neuronales, définies comme des épisodes transitoires d'activité cérébrale suivant des distributions spatiales et temporelles caractéristiques, suggérant un fonctionnement du cerveau à proximité d'un état critique (BEGGS & PLENZ, 2003 ; HAHN et al., 2010 ; PLENZ et al., 2021). La topologie du réseau neuronal apparaît comme un facteur déterminant dans la régulation de ces dynamiques, les patterns de connectivité influençant directement la propagation et la stabilité des avalanches (BORNHOLDT & RÖHL, 2003). Toute altération de cet équilibre critique peut être à l'origine de dysfonctionnements neurologiques majeurs, tels que les crises épileptiques (MEISEL et al., 2012). Ainsi, le bon fonctionnement des systèmes neuronaux dépend fondamentalement de leur robustesse structurelle, qui assure la résilience face à de telles perturbations.

De même, d'autres systèmes complexes, tels que les systèmes financiers et les réseaux électriques, dépendent également de leur robustesse structurelle. Les systèmes financiers présentent des dynamiques de prix

endogènes et des krachs boursiers, reflétant l'état critique et les interactions réseau des participants au marché (BIONDO et al., 2015). Les systèmes électriques révèlent comment les pannes en cascade et les blackouts peuvent être déclenchés et atténués en comprenant les états critiques et la structure du réseau (SHENGWEI MEI et al., 2008). Dans ces deux contextes, le bon fonctionnement de ces systèmes dépend fortement de leur topologie réseau.

3.2 Le tas de sable

La criticalité auto-organisée décrit un mécanisme universel par lequel les systèmes complexes évoluent naturellement vers un état critique, à la frontière entre stabilité et chaos (BAK et al., 1988). Dans cet état, de petites perturbations peuvent entraîner des événements en cascade de différentes amplitudes, suivant une distribution en loi de puissance. La SOC a été observée dans une large gamme de systèmes naturels et artificiels, notamment les tremblements de terre, les feux de forêt, l'activité neuronale et les marchés financiers. Ces systèmes n'ont pas besoin d'être finement ajustés pour atteindre la criticalité ; au contraire, ils s'autorégulent par des dynamiques intrinsèques, faisant de la SOC un cadre convaincant pour comprendre certains comportements émergents dans les systèmes complexes.

Le modèle le plus emblématique illustrant la SOC est le tas de sable de Per Bak, Chao Tang et Kurt Wiesenfeld (BAK et al., 1987), que nous aborderons en premier lieu. L'objectif des auteurs est de proposer une modélisation simple de la SOC afin d'en étudier les dynamiques. Ce modèle est considéré comme classique en raison de sa capacité à démontrer comment des systèmes complexes peuvent évoluer vers un état critique stable.

Plusieurs variantes ont depuis été proposées, notamment le modèle du tas de sable dissipatif. Cette version, que nous aborderons par la suite, explore d'autres dimensions du phénomène, en particulier l'impact des modifications de la topologie sous-jacente du système. Étudier comment ces changements influencent la propagation des avalanches permet d'ouvrir de nouvelles pistes de réflexion sur la résilience et l'adaptabilité des systèmes soumis à une criticalité auto-organisée.

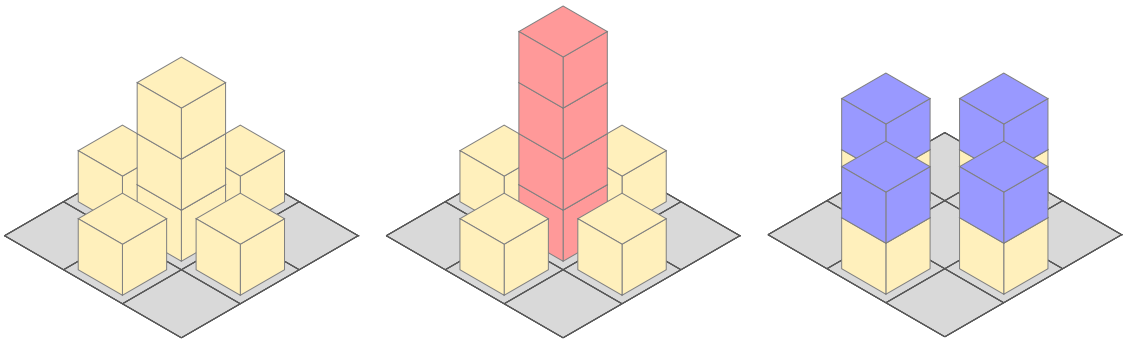
Enfin, nous explorerons quelques autres modèles présentant des caractéristiques de SOC, comme la modélisation de tremblements de terre, de feux de forêt, ou encore d'avalanches neuronales, pour compléter cette section. Ces modèles, bien que différents dans leur approche et ayant leur propre objectif, partagent un but commun de mieux comprendre les mécanismes sous-jacents à l'auto-organisation critique.

3.2.1 Le modèle initial de Bak-Tang-Wiesenfeld

Le modèle canonique proposé par Bak, Tang, et Wiesenfeld (BAK et al., 1987) est un automate cellulaire défini sur une grille régulière de dimension 2, généralement avec une configuration de voisinage de von Neumann, où chaque cellule interagit uniquement avec ses voisins immédiats. Dans cette configuration, chaque cellule (ou nœud) de la grille peut contenir un certain nombre de "grains" qui peuvent "basculer" vers des sites voisins lorsqu'un certain seuil est dépassé. La grille elle-même peut être représentée comme un graphe, où

les nœuds correspondent aux cellules individuelles et les arêtes représentent les connexions entre les cellules voisines. La plupart des cellules ont quatre voisins dans une configuration de von Neumann, à l'exception de celles situées sur les bords de la grille, qui en ont trois, et des cellules d'angle, qui n'en ont que deux.

La dynamique du tas de sable est régie par un ensemble de règles locales simples qui créent de l'instabilité et génèrent des mécanismes d'autorégulation. Chaque cellule a un seuil critique de quatre grains, ce qui signifie que si une cellule accumule quatre grains ou plus, elle devient instable et "s'éboule". Lors de cet événement d'éboulement, la cellule distribue un grain à chacune de ses quatre voisines. Si une cellule voisine est située en dehors de la grille, le grain tombe hors du système, ce qui fait du tas de sable un système ouvert. Des grains sont ajoutés aléatoirement à la grille, et dès que l'ajout d'un grain fait dépasser le seuil d'une cellule, celle-ci s'éboule, ce qui peut déclencher une réaction en chaîne d'éboulements, connue sous le nom d'avalanche. La Figure 3.1 propose une représentation visuelle de l'éboulement d'une cellule. Un grain est déposé sur une cellule au bord de l'instabilité (3 grains), déclenchant un éboulement qui redistribue ses grains aux cellules voisines.



(a) Configuration initiale. (b) Dépôt d'un grain au centre. (c) Éboulement du centre.

FIGURE 3.1 – Illustration d'un éboulement dans un tas de sable de taille 3×3. (a) La cellule centrale est initialement au bord de l'éboulement. (b) Un grain est déposé dessus, provoquant l'instabilité du système. (c) La cellule s'écroule et redistribue ses grains entre ses voisines.

Avec le temps, le système évolue vers un état critique où des avalanches de tailles variées se produisent. La distribution de ces tailles suit une loi de puissance correspondant au bruit rose, également appelé bruit $1/f$. Celui-ci se caractérise par une densité spectrale décroissante en $1/f$, ce qui signifie que chaque octave transporte la même puissance (VOSS & CLARKE, 1975) : les basses fréquences (avalanches courtes) sont plus prononcées qu'avec un bruit blanc, mais sans l'excès d'inertie énergétique du bruit rouge. En revanche, le bruit rouge (ou bruit brownien) présente une densité spectrale décroissante en $1/f^2$, conduisant à une domination encore plus forte des basses fréquences et à un signal de type marche aléatoire, comme décrit dans la littérature sur les spectres de "red noise" (GILMAN et al., 1963).

Ce comportement est caractéristique des systèmes présentant une SOC, où les grands événements sont rares, mais les petits événements sont fréquents. Notamment, le modèle du tas de sable présente une invariance d'échelle et une auto-similarité, ce qui signifie que les mêmes motifs statistiques émergent indépendamment de la taille du système ou du niveau de détail. Pendant une avalanche, aucun nouveau grain n'est ajouté tant que le système n'est pas revenu à une configuration stable, garantissant que tous les événements d'éboulement sont

terminés.

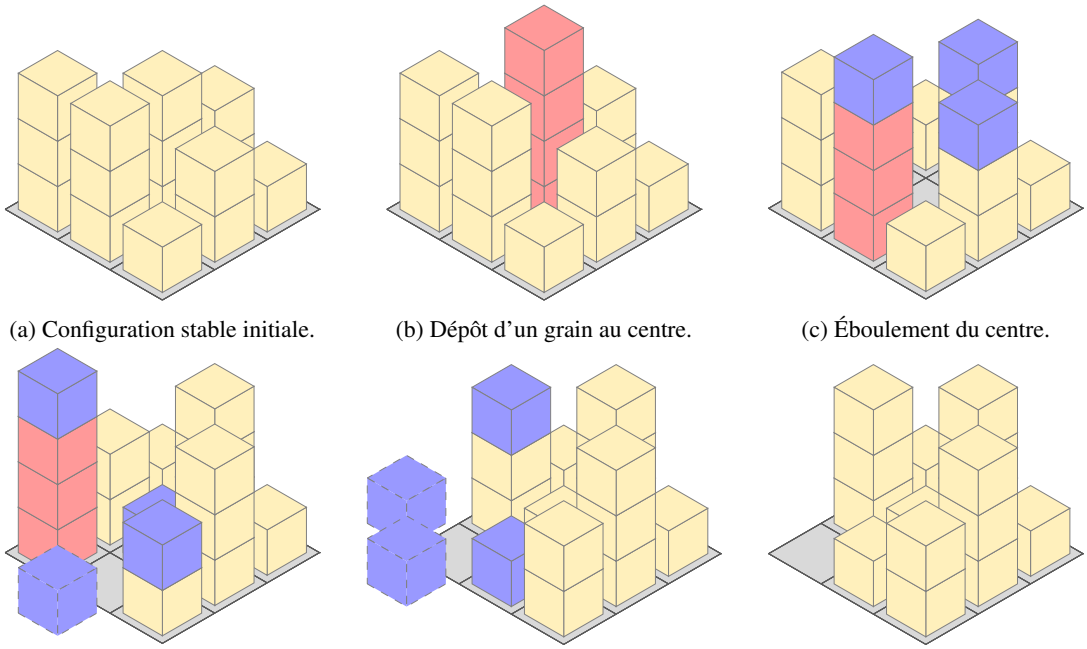
Le processus de simulation du tas de sable est le suivant : un grain est déposé aléatoirement sur la grille à chaque cycle. Si cela rend une cellule instable (c'est-à-dire que le nombre de grains atteint 4), une avalanche de durée indéterminée est déclenchée et résolue avant de passer au cycle suivant. Lors de l'éboulement d'une cellule, un grain est transmis à chacune des cellules voisines, et la cellule en question voit son nombre de grains diminuer de 4. Ainsi, une cellule située en bordure, qui ne dispose pas de quatre voisines, fera disparaître un à deux grains du système pendant son éboulement. Ce mécanisme permet de stabiliser le système en évacuant l'excès de grains. L'Algorithme 1 propose un pseudo-code de ce processus.

Algorithm 1: Processus de simulation du tas de sable canonique

```
   Input: G : grille
          cycles : nombre de cycles de simulation
 1 cycle ← 0
 2 while cycle < cycles do
 3 |   Dépôt d'un grain sur une cellule aléatoire de G
 4 |
   |   /* Gestion de l'avalanche potentielle                    */
 5 |   while au moins une cellule de G est instable do
 6 |   |   foreach cellule de G do
   |   |   |   /* Éboulement de la cellule                      */
 7 |   |   |   if cellule.grains ≥ 4 then
 8 |   |   |   |   foreach voisine de cellule do
 9 |   |   |   |   |   voisine.grains ← voisine.grains + 1
10 |   |   |   |   end
11 |   |   |   |   cellule.grains ← cellule.grains - 4
12 |   |   |   end
13 |   |   end
14 |   end
15 |
16 |   cycle ← cycle + 1
17 end
```

La figure 3.2 illustre un processus typique d'avalanche dans le modèle du tas de sable, démontrant la réaction en chaîne qui se produit lorsqu'un grain est ajouté au système. Comme le montre l'exemple, l'éboulement initial de la cellule centrale se propage à ses voisines, pouvant provoquer une instabilité supplémentaire et entraîner une redistribution plus large des grains. De telles cascades illustrent la dynamique critique du modèle, où de petites perturbations peuvent déclencher des événements à grande échelle. La nature en cascade des avalanches dans le modèle du tas de sable est une caractéristique de la criticalité auto-organisée, mettant en évidence le comportement en loi de puissance et l'invariance d'échelle inhérents à de tels systèmes.

Le modèle de tas de sable BTW est considéré comme abélien car l'ordre dans lequel les événements d'éboulement se produisent durant une avalanche n'affecte pas la configuration finale du système. Cette propriété est essentielle pour simplifier la dynamique du modèle et garantir la reproductibilité des résultats, indépendamment de la séquence des éboulements.



(a) Configuration stable initiale. (b) Dépôt d'un grain au centre. (c) Éboulement du centre.

(d) Éboulement supplémentaire. (e) Dernier éboulement. (f) Configuration stable finale.

FIGURE 3.2 – Exemple d'une avalanche en trois étapes dans un tas de sable BTW de taille 3×3. Un grain est ajouté à la cellule centrale dans la configuration stable initiale (a), ce qui la fait atteindre un seuil critique de 4 (b) et déclenche le début d'une avalanche. Les 4 grains de la cellule centrale sont redistribués à ses voisines (c), provoquant une instabilité supplémentaire et éjectant un grain hors du système (d). L'avalanche se poursuit (e) pour atteindre un nouvel état d'équilibre (f) après l'éjection de deux grains supplémentaires.

3.2.2 Le tas de sable dissipatif

Le modèle dissipatif du tas de sable conserve les mécanismes fondamentaux du fonctionnement original, mais diffère dans la manière dont les grains sortent du système. Contrairement au modèle classique où les grains disparaissent lorsqu'ils tombent en dehors des limites du système depuis un nœud en bordure, les bords du système sont ici fermés. Les grains disparaissent progressivement au fil de leurs déplacements dans le système (BHAUMIK & SANTRA, 2013 ; GOH et al., 2003 ; MALCAI et al., 2006). À chaque transfert d'un grain d'un nœud à un autre pendant une avalanche, il existe une probabilité ϵ que le grain disparaisse. Le modèle dissipatif proposé dans (BHAUMIK & SANTRA, 2013) introduit une contrainte supplémentaire : la dissipation ne peut se produire que lors de transferts vers des nœuds aléatoires préalablement sélectionnés, représentant ainsi les bordures du modèle canonique. Bien que cette probabilité ϵ puisse être définie de diverses manières, elle est généralement ajustée et fixée à l'avance, car elle influence directement la tension du système nécessaire pour atteindre la SOC. En effet, la probabilité doit permettre au système d'atteindre un état stationnaire, caractérisé par l'équilibre entre le nombre de grains ajoutés et ceux dissipés. Le mécanisme de dissipation rend ces modèles non conservatifs, en opposition au modèle canonique.

L'introduction de la dissipation dans les modèles de tas de sable permet de mieux comprendre les phénomènes naturels où la conservation stricte de la matière ou de l'énergie n'est pas respectée. Par exemple, dans les systèmes géophysiques tels que les tremblements de terre ou les glissements de terrain, une partie de l'énergie est dissipée sous forme de chaleur ou d'autres formes non récupérables. En biologie, ces modèles peuvent

être utilisés pour étudier la propagation des signaux neuronaux, où certains signaux peuvent être perdus ou atténués au cours de leur transmission. Un autre avantage des modèles dissipatifs est leur capacité à reproduire des comportements critiques plus réalistes. En ajustant la probabilité de dissipation ϵ, il est possible de moduler la dynamique du système pour qu'il atteigne un état de SOC plus stable et robuste. Cela permet de mieux comprendre comment les systèmes naturels et artificiels peuvent s'adapter et évoluer face à des perturbations externes. En outre, les modèles dissipatifs permettent d'explorer les transitions de phase et les points critiques dans les systèmes non-conservatifs. En variant la probabilité de dissipation, on peut observer des changements dans les propriétés statistiques des avalanches, telles que leur taille et leur durée. Ces observations peuvent fournir des informations précieuses sur la manière dont les systèmes complexes réagissent aux perturbations et aux contraintes externes.

3.2.3 Autres modèles présentant de la SOC

Bien que le modèle du tas de sable en soit le paradigme fondateur, il n'est pas le seul à manifester des comportements relevant de la criticalité auto-organisée. D'autres modèles, qu'ils en soient directement dérivés ou non, présentent également des signatures caractéristiques de la SOC (TURCOTTE, 1999). Nous détaillerons quelques-uns de ces modèles dans cette section en nous intéressant aux modèles de tremblements de terre d'Olami-Feder-Christensen, de percolation (incluant le modèle de feux de forêt), d'avalanches neuronales, et pour terminer le Chip-firing game.

Modèle d'Olami-Feder-Christensen : l'étude statistique des tremblements de terre montre que la croûte terrestre se comporte comme un système complexe et présente des dynamiques s'apparentant à de la SOC (SORNETTE & SORNETTE, 1989), notamment par la présence de multiples lois de puissance dans les mesures. Les auteurs de (OLAMI et al., 1992) qualifient les tremblements de terre de "paradigme le plus pertinent de la criticalité auto-organisée". Les recherches se sont donc naturellement orientées vers la modélisation des séismes par des mécanismes de SOC. Olami, Feder et Christensen ont proposé une modélisation (OLAMI et al., 1992) (modèle OFC) basée sur une version simplifiée du modèle ressort-bloc de Burridge-Knopoff (BURRIDGE & KNOPOFF, 1967), qui modélise les tremblements de terre. Ce modèle est un automate cellulaire où chaque cellule représente un bloc, les cellules étant reliées par des ressorts. Elles accumulent de l'énergie jusqu'à un seuil critique avant d'effectuer un "glissement", exerçant une force sur le voisinage via les ressorts. Les voisins peuvent à leur tour dépasser le seuil critique et contribuer à la propagation, provoquant une avalanche de réactions. Les interactions inter-cellules sont régies par des paramètres, notamment l'élasticité des ressorts. Le modèle OFC incorpore une dissipation de l'énergie, rendant le modèle non-conservatif et plus réaliste. Les auteurs montrent que l'exposant des lois de puissance observées est directement corrélé au coefficient de dissipation.

Modèles de percolation et de feux de forêt : les modèles de percolation sont généralement utilisés pour modéliser et étudier la connectivité et la propagation dans des systèmes désordonnés. Ils s'appliquent à divers phénomènes, tels que la dynamique des fluides dans des milieux poreux, la conduction électrique dans des

matériaux composites, ou encore la propagation des feux de forêts. Le principe de ces modèles repose sur la génération d'un réseau (souvent une grille) où chaque nœud, appelé *site*, peut être occupé ou non selon une probabilité. Au fil de la simulation, les sites changent d'état en fonction de leur voisinage. Les études se concentrent principalement sur la formation de clusters de sites occupés lorsque l'état de certains sites est modifié. Il existe un seuil critique de densité d'occupation, appelé seuil de percolation, au-delà duquel un cluster infini apparaît, signifiant une propagation de l'occupation à travers tout le système. Certaines modélisations intègrent des mécanismes qui ajustent dynamiquement la probabilité d'occupation des sites pour amener spontanément le système à proximité du seuil de percolation. C'est notamment le cas du modèle de percolation auto-organisée (Self-Organized Percolation) (ALENCAR et al., 1997), dont les auteurs comparent le comportement au concept de SOC. Le modèle de percolation dirigée (Directed Percolation) (VÁZQUEZ & COSTA, 1999) est également lié à la SOC. Il intègre une notion de direction dynamique pour la propagation, ainsi que des états absorbants. Ces états apparaissent spontanément en fonction d'une probabilité de propagation des avalanches, paramètre du système, permettant à celles-ci d'être finies. Une faible probabilité conduit tous les sites vers une inactivité définitive, tandis qu'une probabilité suffisamment élevée peut entraîner une propagation continue.

Les modèles de percolation sont également utilisés dans l'étude des feux de forêt (DROSSEL & SCHWABL, 1992) à travers une extension du modèle classique de percolation de sites, incluant un mécanisme spécifique de déclenchement d'incendies. À chaque pas de temps, un arbre peut être placé aléatoirement sur un site vacant et, à intervalle régulier, une étincelle est déposée aléatoirement sur un arbre, provoquant ainsi un départ d'incendie. Le feu ainsi déclenché se propage aux arbres situés sur les sites voisins, modifiant leur état en "en feu" et générant un feu de forêt, réaction en chaîne assimilable à une avalanche. Cette propagation se poursuit jusqu'à épuisement des arbres dans le voisinage immédiat du front d'incendie. Les arbres brûlés disparaissent alors, laissant leurs sites vides. Grâce au processus continu d'ajout (plantation aléatoire) et de retrait (incendies) d'arbres, le système s'auto-organise spontanément dans un état où la taille des feux suit une distribution en loi de puissance.

Modèles d'avalanches neuronales : les réseaux neuronaux ont été largement étudiés au cours des dernières décennies, et ils présentent des dynamiques de SOC résultant du mécanisme d'activation en cascade des neurones (HESSE & GROSS, 2014 ; PLENZ et al., 2021). Plusieurs travaux, tels que (de ARCANGELIS et al., 2006 ; RYBARSCH & BORNHOLDT, 2014), proposent des modélisations de réseaux neuronaux pour capturer et analyser l'essence naturelle de la SOC dans ces réseaux.

Les auteurs de (de ARCANGELIS et al., 2006) ont développé un modèle reproduisant la plasticité cérébrale. Ce modèle consiste en un réseau électrique où les nœuds représentent des neurones et les liaisons, les synapses entre neurones voisins. Chaque neurone possède un potentiel qui, lorsqu'il dépasse un seuil, est déchargé vers les voisins proportionnellement au potentiel de chacune de ses liaisons. Un neurone ainsi déclenché retrouve un potentiel nul et devient temporairement inactif, n'acceptant aucune charge de ses voisins. L'état des liaisons est mis à jour à chaque pas de temps pour augmenter leur capacité lorsqu'elles sont utilisées et la réduire lorsqu'elles ne le sont pas, jusqu'à atteindre une capacité nulle. Ces évolutions reproduisent le renforcement et l'affaiblissement synaptique du cerveau, pouvant mener à la disparition des liaisons synaptiques. Les résultats

obtenus avec ce modèle montrent les lois de puissance caractéristiques de la SOC. Les auteurs de (RYBARSCH & BORNHOLDT, 2014) proposent un modèle minimal de réseaux neuronaux, visant à servir de base solide pour des modèles plus complexes. Cette modélisation s'inspire des modèles de spins, dont le modèle d'Ernst-Ising est le plus connu : des électrons sont mis en réseau et disposent d'une orientation (spin : $\{\frac{1}{2}; -\frac{1}{2}\}$) ; l'état suivant d'un électron dépend de son voisinage ainsi que de paramètres externes tels que la température. Dans le modèle de réseau neuronal de (RYBARSCH & BORNHOLDT, 2014), les neurones (nœuds du réseau) possèdent un état booléen, et leur état à l'étape suivante dépend de l'état moyen de leur voisinage comparé à un seuil critique d'activation. Ce seuil est maintenu localement au cours du temps : à chaque pas de temps, un neurone a une probabilité d'augmenter son seuil égale à la moyenne de ses activations sur les V derniers pas de temps, V correspondant au temps d'épuisement des neurones. Ce modèle présente les signatures de la SOC, et les résultats correspondent aux données expérimentales sur l'activité corticale.

Chip-firing game : le Chip-firing game (BJÖRNER et al., 1991 ; MERINO, 2005) est un jeu solitaire où des jetons sont initialement empilés sur les sommets d'un graphe. Une étape du jeu consiste à sélectionner un sommet possédant au moins autant de jetons que son degré et à faire "tirer" un jeton vers chacun de ses voisins, réduisant ainsi son nombre de jetons d'autant que son degré. Le jeu s'achève lorsqu'aucun sommet ne dispose plus d'assez de jetons pour être "tiré". Une partie de Chip-firing game, telle qu'illustré par la Figure 3.3, peut être comparée à une avalanche dans un modèle de tas de sable (Figure 3.2). De plus, comme le tas de sable, le Chip-firing game est abélien : une disposition initiale des jetons conduira invariablement à une même disposition finale, indépendamment de l'ordre dans lequel les sommets sont "tirés". En adoptant une perspective plus large, le tas de sable peut être considéré comme une configuration spécifique du Chip-firing game, où tous les sommets ont un degré de 4. Cependant, contrairement au tas de sable, qui est un système ouvert permettant l'évacuation de l'excès de grains, le Chip-firing game est un système fermé. Par conséquent, si le nombre initial de jetons est trop élevé, la partie peut devenir infinie. La similarité des mécanismes du Chip-firing game avec ceux du tas de sable en fait un modèle présentant des caractéristiques de SOC.

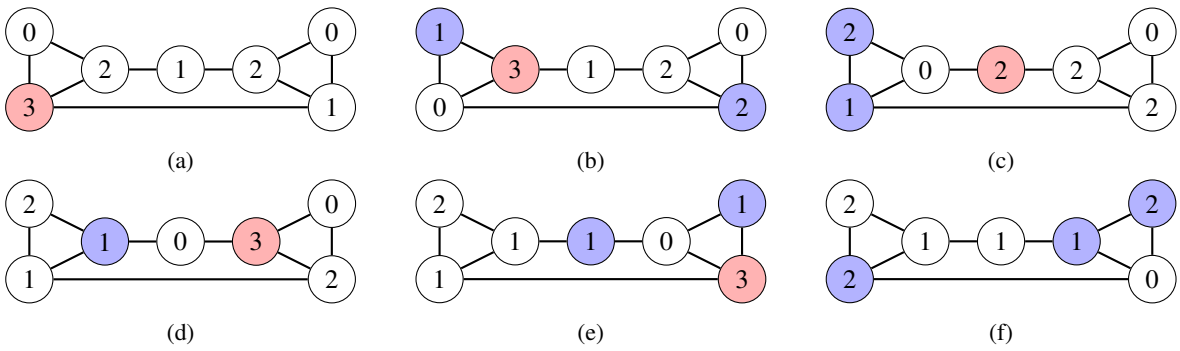


FIGURE 3.3 – Exemple d'une partie de Chip-firing game. À chaque étape, un nœud disposant d'au moins autant de jetons que de voisins est sélectionné (nœud rouge). Il "tire" alors un jeton vers chaque voisin (nœuds bleus). Ce processus est répété jusqu'à ce qu'aucun nœud ne détienne plus d'assez de jetons pour être sélectionné.

3.3 Topologies de réseau dans les systèmes SOC

Alors que le modèle classique du tas de sable de Bak-Tang-Wiesenfeld est généralement implémenté sur une grille régulière, de nombreux systèmes réels présentant une SOC se produisent dans des réseaux complexes (BASSETT & BULLMORE, 2006). Contrairement aux grilles régulières, les réseaux complexes englobent une large gamme de structures caractérisées par des motifs de connectivité différents, y compris des topologies de type petit-monde et sans échelle. Ces variations dans la structure du réseau peuvent avoir un impact significatif sur l'émergence et les propriétés statistiques de la SOC, motivant des recherches approfondies sur la manière dont différentes topologies influencent les dynamiques de la SOC (BHAUMIK & SANTRA, 2013 ; de ARCANGELIS & HERRMANN, 2002 ; GOH et al., 2003 ; KARMAKAR & MANNA, 2005 ; PAN et al., 2007).

Une approche notable consiste à transformer une grille régulière en un réseau de type petit-monde par recâblage. Dans (BHAUMIK & SANTRA, 2013), une méthode de reconnexion est introduite, des arêtes aléatoires sont ajoutées entre des paires de nœuds, modifiant ainsi la connectivité du réseau. Pour s'adapter à ce changement structurel, le mécanisme d'expulsion des grains basé sur les bords est remplacé par un modèle de dissipation tel que présenté en Section 3.2.2. En revanche, la dissipation proposée par BHAUMIK et SANTRA (2013) ne s'effectue qu'à partir de nœuds sélectionnés arbitrairement plutôt que pour n'importe quel déplacement de grain. La probabilité de dissipation choisie correspond à la fréquence empirique d'éjection des grains par les bordures du tas de sable canonique. De même, PAN et al. (2007) étudient les graphes dirigés en modifiant la destination des connexions sortantes, permettant à chaque nœud d'avoir un nombre variable d'arêtes entrantes tout en conservant exactement quatre arêtes sortantes. Cette approche préserve les dynamiques du tas de sable sans nécessiter de modifications supplémentaires. Une autre méthode, décrite par de ARCANGELIS et HERRMANN (2002), incorpore également des ajouts aléatoires d'arêtes mais les compense en supprimant d'autres, garantissant que le degré moyen des nœuds reste à quatre. Malgré les variations de mise en œuvre, les trois études révèlent une évolution commune : à mesure que les réseaux deviennent de plus en plus reconnectés, les distributions en loi de puissance régissant les tailles des avalanches deviennent plus pentues, indiquant un changement dans le comportement critique du système.

Au-delà des réseaux de type petit-monde, les réseaux sans échelle (Scale-Free Networks) (BARABÁSI & ALBERT, 1999) ont également été étudiés dans le contexte de la SOC. Ces réseaux sont caractérisés par une distribution de degrés hétérogène, où quelques nœuds (hubs) ont une connectivité nettement plus élevée que les autres. Cette caractéristique structurelle est couramment observée dans les systèmes naturels et joue un rôle crucial dans les dynamiques de la SOC. Les réseaux sans échelle ont été étudiés dans (GOH et al., 2003). Les structures ont été générées en utilisant l'algorithme proposé dans (GOH et al., 2001), où les arêtes sont ajoutées en fonction des poids des nœuds pour atteindre le degré moyen souhaité. Il est important de noter que le modèle du tas de sable est alors dissipatif pour préserver sa nature de système ouvert. La relation entre l'exposant de degré du réseau sans échelle (γ) et l'exposant de la distribution des tailles d'avalanches (τ) est examinée. L'étude montre que lorsque $2 < \gamma < 3$, l'exposant τ suit la relation $\tau = \dfrac{\gamma}{\gamma - 1}$, tandis que pour $\gamma > 3$, τ converge vers 1,5. La Figure 3.4 illustre ces différentes situations par quatre courbes représentant la distribution des tailles des avalanches pour différents γ. Plus γ est faible, plus τ est élevé. Cela indique que

les réseaux avec une concentration plus élevée de hubs (plus faible γ) présentent des longueurs de chemin plus courtes et des distributions d'avalanches plus pentues, renforçant les observations précédentes des réseaux de type petit-monde.

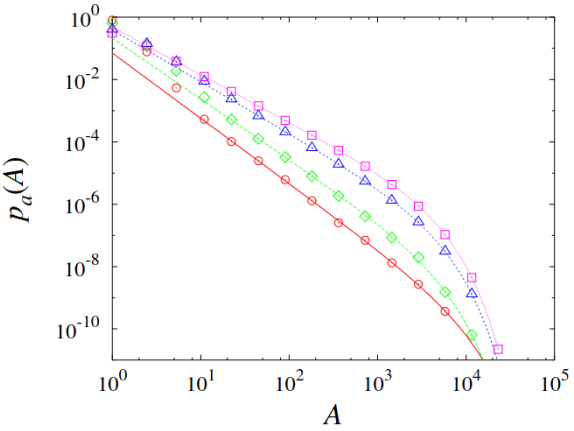


FIGURE 3.4 – Figure 1 des travaux de GOH et al. (2003) illustrant la pente de la distribution des avalanches dans des réseaux sans échelle de différents exposants de degré γ. Plus γ est faible, plus la distribution est pentue : $\gamma = \infty$ (magenta □), $\gamma = 3$ (bleu △), $\gamma = 2{,}2$ (vert ◇), et $\gamma = 2$ (rouge ◯). Lorsque $\gamma \to \infty$, $\tau \to 1{,}5$. La fréquence des avalanches est exprimée en probabilité de parution par rapport à toutes les avalanche survenues.

Une approche alternative est explorée dans (KARMAKAR & MANNA, 2005), où les réseaux sans échelle sont intégrés dans une grille tout en maintenant des dynamiques de tas de sable conservatrices. En modifiant le réseau pour minimiser les longueurs des liens, les réseaux sans échelle optimisés résultants présentent un comportement multi-échelle similaire à celui des grilles à maillage carré. En revanche, les réseaux sans échelle non optimisés ne montrent pas cette caractéristique, soulignant le rôle de la structure du réseau dans la détermination des propriétés de la SOC. Ces résultats montrent collectivement que la topologie sous-jacente a un impact profond sur les dynamiques du tas de sable, influençant non seulement la présence de la SOC mais aussi ses propriétés statistiques.

L'influence de la topologie du réseau sur la SOC ne se limite pas aux modèles de tas de sable. D'autres systèmes de SOC, tels que le modèle de tremblement de terre d'Olami-Feder-Christensen (OLAMI et al., 1992), ont été étudiés sur diverses structures de réseau. Dans (LISE & PACZUSKI, 2002), les graphes aléatoires d'Erdös-Rényi (ERDÖS & RÉNYI, 1960) et les graphes évolutifs sont analysés. La différence essentielle entre ces deux types réside dans la nature statique des graphes d'Erdös-Rényi par rapport à la réaffectation dynamique des voisins dans les graphes évolutifs à chaque étape temporelle. L'étude montre que le comportement de SOC est observé dans le modèle d'Olami-Feder-Christensen sur les graphes d'Erdös-Rényi, avec un exposant de loi de puissance de $\tau \simeq 1{,}65$, indépendamment du fait que le système soit conservatif ou non. Cependant, dans les graphes évolutifs, la SOC n'émerge que dans les systèmes conservatifs. Au final, l'étude suggère que des interactions locales fixes, développant d'une corrélation spatiale, sont essentielles pour maintenir la SOC dans les systèmes non-conservatifs (dissipatifs).

Un exemple particulièrement significatif de SOC dans un système en réseau est l'architecture neuronale du cerveau. Des études expérimentales ont montré que les réseaux neuronaux dans les couches superficielles

du cerveau développent des motifs de connectivité préférentiels, formant souvent des structures ressemblant à des réseaux sans échelle (BEGGS & PLENZ, 2003 ; EYTAN & MAROM, 2006 ; HAHN et al., 2010 ; PLENZ et al., 2021). Ces réseaux contiennent des neurones "hubs" hautement connectés, qui s'activent avant le reste du réseau, déclenchant des avalanches neuronales, une caractéristique de la SOC dans le cerveau. Notamment, ces avalanches n'émergent qu'une fois le réseau complètement mature. En revanche, les systèmes neuronaux dépourvus de cette organisation préférentielle tendent à fonctionner dans un état supercritique, où l'activité se propage de manière incontrôlable plutôt que de s'autoréguler. De plus, il a été montré que l'équilibre entre les neurones excitateurs et inhibiteurs est crucial pour maintenir le comportement de SOC dans le cerveau (EYTAN & MAROM, 2006).

Ces études témoignent du rôle fondamental de la topologie du réseau dans la détermination des dynamiques des systèmes SOC. Qu'il s'agisse de modèles de tas de sable, de simulations de tremblements de terre ou de réseaux neuronaux biologiques, la structure sous-jacente des interactions influence non seulement l'émergence de la SOC mais aussi son maintien face à des variations de structure. Comprendre ces effets fournit des informations sur le caractère transversal de la SOC à travers divers systèmes et offre un cadre pour analyser l'impact des changements structurels dans les réseaux complexes du monde réel.

3.4 Robustesse des systèmes SOC

La robustesse des systèmes SOC peut être analysée selon deux perspectives complémentaires : d'une part, les propriétés structurelles de la topologie du réseau sous-jacent, et d'autre part, les processus internes régissant la dynamique du système. Ces deux aspects jouent un rôle crucial dans le maintien de la stabilité tout en préservant l'état critique nécessaire au comportement de la SOC.

3.4.1 Robustesse structurelle : organisations hiérarchiques et modulaires

Une stratégie courante pour renforcer la robustesse des systèmes complexes consiste à intégrer des structures modulaires hiérarchiques. Dans ces réseaux, les nœuds ayant des interactions fréquentes sont regroupés en modules, où les connexions intra-modules sont plus denses que les connexions inter-modules. L'aspect hiérarchique apparaît lorsque les modules eux-mêmes forment des clusters de niveau supérieur, créant une structure multi-couches qui équilibre la coordination locale et globale.

Ce type d'organisation a été largement étudié dans les systèmes biologiques, en particulier dans les réseaux neuronaux (MEISEL et al., 2012 ; S.-J. WANG & ZHOU, 2012). Le cerveau, par exemple, exploite la modularité hiérarchique pour garantir à la fois l'efficacité et la résilience. Cette structure empêche les perturbations locales de se propager de manière incontrôlée, évitant ainsi les défaillances à grande échelle telles que celles observées lors des crises épileptiques (MEISEL et al., 2012), où le système passe d'un état de SOC à un régime supercritique caractérisé par une propagation incontrôlée de l'activité. Cependant, une structure excessivement modulaire peut entraver la transmission globale de l'activité, risquant de pousser le système vers un état sous-critique. Pour maintenir la criticalité, la modularité est couplée à une adaptabilité structurelle, un mécanisme

également observé dans les systèmes neuronaux (MEISEL et al., 2012).

3.4.2 Robustesse dynamique : auto-adaptation et mécanismes de contrôle

Au-delà des considérations structurelles, la robustesse des systèmes SOC provient également de mécanismes auto-adaptatifs qui régulent la dynamique du système. Les études sur les réseaux adaptatifs (GROSS & BLASIUS, 2008 ; ROHLF & BORNHOLDT, 2009) montrent que la SOC peut émerger dans des systèmes où la connectivité évolue au fil du temps en fonction des interactions entre les nœuds. Ces réseaux présentent un haut degré de résilience, car l'auto-organisation réduit la dépendance aux conditions initiales et aux paramètres externes.

Une autre approche pour améliorer la robustesse consiste à contrôler la dynamique des avalanches afin de prévenir les effondrements à l'échelle du système (CAJUEIRO & ANDRADE, 2010). Une stratégie consiste à déclencher de manière préventive des avalanches locales dans les régions proches de l'état critique, empêchant ainsi une accumulation excessive d'énergie qui pourrait conduire à des cascades à grande échelle. Deux méthodes de sélection sont proposées pour déclencher ces avalanches préventives :

1. Sélection basée sur le degré : les nœuds ayant la plus forte connectivité sont surveillés, et lorsque l'un d'eux approche du seuil critique, une avalanche est déclenchée. Cette méthode est très efficace mais nécessite une connaissance complète de la structure du réseau.
2. Sélection aléatoire : les nœuds sont choisis aléatoirement pour être surveillés et faire l'objet d'interventions, offrant une stratégie plus réalisable lorsque les informations sur l'ensemble du réseau ne sont pas disponibles, bien que légèrement moins efficace.

Ces résultats soulignent l'interaction entre la topologie du réseau et les mécanismes internes de régulation dans le maintien de la SOC. Alors que la modularité hiérarchique assure une résilience structurelle, la connectivité adaptative et le déclenchement contrôlé des avalanches garantissent une stabilité dynamique, empêchant le système de dériver vers des états supercritiques ou sous-critiques.

3.5 Le tas de sable pour de l'équilibrage dynamique

Maintenant que nous avons exploré la criticalité auto-organisée et touché du doigt son potentiel, il est temps de revenir sur notre sujet principal : l'équilibrage de charge. Il est tentant d'imaginer un système d'équilibrage présentant une telle dynamique pour auto-organiser de manière décentralisée les tâches entre ses ressources. La question est alors : comment modéliser un système d'équilibrage présentant des dynamiques de SOC ?

Bien que le tas de sable constitue un cadre conceptuel très important, il ne reflète pas, malheureusement, le fonctionnement réel d'un système de traitement de tâches. Si une analogie peut être faite entre les grains qui se déplacent et les tâches cherchant une ressource disponible, le mécanisme d'expulsion des grains hors du système (autorisant le retour à un état stable) n'est pas transposable à un système de traitement de tâches, dans lequel chacune doit impérativement être prise en charge.

Afin de mieux représenter ce type de système, deux extensions du modèle initial ont été proposées dans des travaux antérieurs à cette thèse par LAREDO et al. (2012, 2014) : le tas de sable ordonnanceur et le tamis. Ces deux modèles modifient le principe d'évacuation en introduisant une ouverture en surface, permettant un traitement progressif des grains. Nous verrons que ces deux modèles démontrent des performances d'ordonnancement au moins équivalentes, voire supérieures, à celles des solutions traditionnelles sur certains aspects. Les études menées sur ces modèles soulignent le potentiel des modèles de type tas de sable pour gérer efficacement la charge de travail dans des environnements dynamiques, tout en maintenant une qualité de service élevée et une consommation énergétique optimisée.

Dans cette section, nous allons tout d'abord nous intéresser au tas de sable ordonnanceur. Puis, nous présenterons un système d'équilibrage et d'ordonnancement inspiré du modèle du tas de sable et du modèle précédent. Enfin, nous terminerons cette section et ce chapitre par l'étude du modèle du tamis, sur lequel se basent les travaux présentés au Chapitre 5.

3.5.1 Le tas de sable ordonnanceur

Le tas de sable ordonnanceur (Sandpile Scheduler) a été introduit dans (LAREDO et al., 2012) puis amélioré dans (LAREDO et al., 2014), dans le but de proposer une solution d'équilibrage de charge dynamique adaptée à des environnements décentralisés.

Ce modèle vise à explorer dans quelle mesure les dynamiques issues de la criticalité auto-organisée peuvent être exploitées pour produire un équilibrage de charge efficace, en s'inspirant des mécanismes fondamentaux du modèle de tas de sable. Toutefois, ce dernier ne permet pas de simuler fidèlement un système de traitement de tâches, ce qui justifie l'introduction de deux modifications majeures dans le tas de sable ordonnanceur : une nouvelle approche du mécanisme d'évacuation des grains, et une gestion adaptée des avalanches.

Les grains ne peuvent plus être évacués hors du système par les bordures. Chaque cellule est désormais dotée d'une capacité de traitement, ce qui permet une prise en charge progressive des grains au fil du temps. De plus, la taille des grains devient variable, introduisant ainsi une granularité plus fine dans la modélisation de la charge de travail. Ce nouveau modèle permet de faire un parallèle direct entre le tas de sable ordonnanceur et un système de traitement de tâches, dont le Tableau 3.1 propose les correspondances.

Tas de sable ordonnanceur		**Système de traitement de tâches**
Grain de sable	$\Leftrightarrow$	Tâche à traiter
Taille de grain	$\Leftrightarrow$	Durée de la tâche
Cellule de l'automate	$\Leftrightarrow$	Unité de calcul
Vitesse de traitement	$\Leftrightarrow$	Vitesse de calcul
Grains sur une cellule	$\Leftrightarrow$	File d'attente de l'unité de calcul

TABLE 3.1 – Correspondances entre le tas de sable ordonnanceur et un système de traitement de tâches.

Chaque cellule de l'automate est dotée d'un agent dont le rôle est de surveiller la charge de la cellule correspondante ainsi que celle des cellules voisines. Lorsque la quantité de grains présente sur une cellule dépasse le nombre total des grains dans son voisinage, l'agent déclenche un écroulement de la cellule et réaffecte des

grains à ses voisines. L'une des voisines peut alors devenir instable à son tour, prolongeant ainsi l'avalanche. Ce mécanisme, en plus de reproduire les dynamiques de la SOC, permet un équilibrage efficace des grains dans le système.

La notion de bordure n'étant plus pertinente en raison du nouveau paradigme d'évacuation des grains, le tas de sable ordonnanceur a été étudié sur des structures de graphe petit-monde générées à partir d'un maillage en anneau. Ces structures permettent d'explorer des configurations plus réalistes et adaptées aux systèmes de traitement de tâches modernes.

Le tas de sable ordonnanceur a été testé avec le problème d'ordonnancement de sacs-de-tâches (Bags-of-Tasks scheduling problem) (IOSUP et al., 2008b), où des ensembles de tâches à traiter en parallèle sont envoyés dans le système. Le modèle démontre de meilleures performances (temps de traitement total, quantités de traitements, nombre de migrations des tâches) sur des structures de type petit-monde que sur des maillages, avec des résultats proches de l'optimalité. Cela s'explique par la capacité des avalanches à atteindre des zones plus distantes de la structure pour disperser la charge, tandis que les avalanches restent locales dans un maillage. De plus, le modèle démontre une excellente capacité à gérer la charge dans des environnements hétérogènes, où les ressources (cellules) disposent de vitesses de traitement différentes. Enfin, le tas sa sable ordonnanceur présente de meilleures performances que des systèmes d'équilibrage classiques comme l'affectation aléatoire ou le tourniquet (Round Robin).

En outre, le modèle a été amélioré avec un protocole de bavardage (Gossip protocol) visant à réduire le nombre de migrations des tâches, impactant ainsi la consommation énergétique du système. Plutôt que de transférer physiquement les tâches durant les avalanches, seules quelques informations nécessaires sont transmises. Une fois une configuration stable trouvée, les tâches sont effectivement déplacées vers leur ressource finale de l'avalanche. Ce mécanisme permet de réduire considérablement l'impact énergétique lié au déplacement des tâches pour équilibrer le système.

3.5.2 Un ordonnanceur et équilibreur de charge décentralisé

Le système proposé par GĄSIOR et SEREDYŃSKI (2017) constitue une approche décentralisée pour l'ordonnancement et l'équilibrage de tâches dans un environnement de cloud computing, s'inspirant des dynamiques du modèle de tas de sable et du tas de sable ordonnanceur. Ces travaux visent principalement à optimiser les performances du système en réduisant le temps d'exécution des tâches. L'environnement considéré est statique et entièrement connu à l'avance : le nombre de ressources de calcul, leur puissance, le nombre de cœurs disponibles, ainsi que les latences de communication entre ressources sont fixés.

Afin de distribuer les tâches au mieux, un mécanisme local aux nœuds déclenche la répartition des tâches à l'instar du modèle du tas de sable. Cependant, la décision du déclenchement résulte d'une comparaison de l'état local d'un nœud à son voisinage et non pas de son propre état uniquement, comme le tas de sable ordonnanceur. Chaque nœud dispose d'une estimation de sa charge comparée au voisinage, déterminant son état : surchargé, équilibré ou sous-chargé. Cette estimation est découpée en deux indicateurs : le temps de complétion maximal des tâches en attente (C_{max}) et le temps de calcul non utilisé (τ). Le premier indicateur

correspond à l'accumulation de tous les temps de complétion des tâches d'un nœud. Le second correspond, suite à un ordonnancement local des tâches, au temps de calcul du nœud non utilisé par les tâches jusqu'à leur complétion totale. Un τ à 0 signifie qu'aucune nouvelle tâche ne peut être ordonnancée sans violer les dates limites de tâches présentes. Ainsi, un nœud est surchargé lorsque la différence entre C_{max} et $\overline{C_{max}^{voisins}}$ dépasse un seuil fixé préalablement ou lorsque $\tau = 0$. Dans cet état, toute nouvelle tâche arrivant sur le nœud est automatiquement redirigée vers un voisin, de même que les tâches locales ne pouvant être traitées localement dans le temps imparti. Un des nœuds voisins recevant l'une des tâches en excès peut alors devenir lui-même surchargé provoquant une réaction en chaîne : une avalanche.

Bien que la consommation énergétique constitue un enjeu majeur dans les environnements de cloud computing, elle n'est pas prise en compte dans ces travaux. De même, la problématique de la surcharge globale du système est ignorée, alors même qu'une telle situation peut survenir à tout moment. Dans le cadre du mécanisme de répartition proposé, une surcharge implique un fonctionnement intensif : les tâches continuent de se déplacer dans le système jusqu'à ce qu'un état stable soit atteint, où aucun nœud n'est surchargé. Ce processus peut nécessiter une avalanche de grande ampleur, entraînant une consommation énergétique significative.

3.5.3 Le tamis

Le modèle du tamis a été introduit dans (LAREDO et al., 2017), à l'instar du tas de sable ordonnanceur, afin d'examiner dans quelle mesure les dynamiques de criticalité auto-organisée peuvent favoriser un équilibrage de charge efficace, en s'inspirant des mécanismes du tas de sable. La principale différence entre ces deux modèles réside dans le fait que le tamis conserve les mécanismes du modèle canonique d'automate cellulaire, et se différencie également par le type d'environnement considéré pour le système.

Le tamis introduit une ouverture du système en surface, similaire au tas de sable ordonnanceur : les cellules disposent d'une capacité de tamisage (vitesse de traitement) et les grains d'une taille donnée. Ces derniers sont tamisés progressivement au fil du temps par les cellules. Il est donc possible d'établir un parallèle entre le tamis et un système de traitement, dont certaines correspondances ont été proposées précédemment dans le Tableau 3.1. L'Algorithme 2 présente le fonctionnement du tamis en pseudo-code. À chaque cycle, en plus des étapes classiques du tas de sable (dépôt d'un grain et gestion de l'instabilité), le tamis ajoute une étape de tamisage.

La Figure 3.5 illustre le modèle du tamis. Les grains sont déposés sur la grille, de manière similaire au modèle du tas de sable. Cependant, si un grain se retrouve sur une cellule vide (par un dépôt ou une avalanche), il entre en état de tamisage et ne participe plus aux mécanismes canoniques de l'automate cellulaire. Selon sa taille et la capacité de tamisage de la cellule, le grain sera progressivement tamisé et finira par sortir du système (par le dessous de la grille). Par ce mécanisme, les grains passent progressivement au travers du système, tel que le ferait du sable dans un tamis réel.

Ce nouveau modèle introduit un paramètre crucial : la capacité de tamisage des cellules. À l'instar de la probabilité de dissipation dans les modèles dissipatifs du tas de sable (Section 3.2.2), la capacité de tamisage influence directement la dynamique des avalanches, notamment lorsque le système est de taille finie. En effet,

Algorithm 2: Processus de simulation du tamis

```
Input: G : grille
       cycles : nombre de cycles de simulation
cycle ← 0
while cycle < cycles do
    Étape 1 : dépôt d'un grain sur une cellule aléatoire de G
    Étape 2 : gestion de l'avalanche potentielle

    /* Tamisage des grains                                          */
    foreach cellule ∈ G do
        if cellule non vide then
            Progression du tamisage des grains de la cellule
        end
    end

    cycle ← cycle + 1
end
```

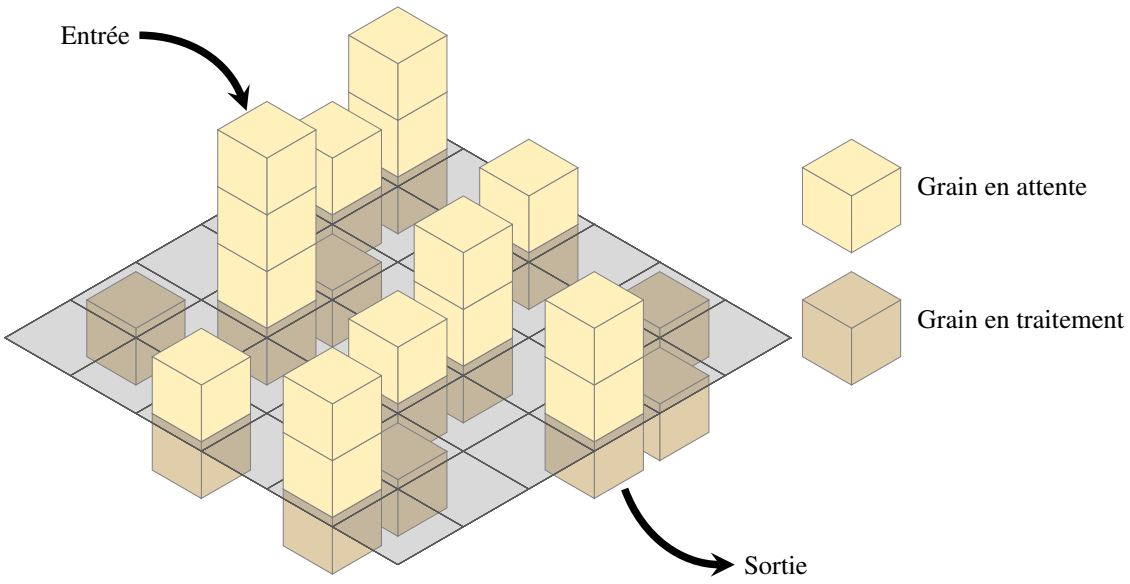


FIGURE 3.5 – Illustration du modèle du tamis. Les grains clairs jouent le même rôle que dans le tas de sable canonique, tandis que les grains foncés, en train d'être tamisés, n'influent plus sur les avalanches. L'ouverture du système se fait par le tamisage progressif des grains.

le traitement des grains s'effectuant au fil du temps (et non pendant les avalanches), plus la charge de grains (nombre et/ou taille) augmente, plus des avalanches de très grande échelle surviendront. Lorsque la charge atteindra presque la capacité totale de tamisage du système par cycle, aucune configuration stable ne pourra plus être atteinte. La capacité de tamisage joue donc un rôle déterminant dans la gestion de la charge et de la stabilité globale du système. Dans le contexte d'un système de traitement de tâches, une capacité de tamisage bien calibrée permet de réguler le flux de tâches et d'éviter les surcharges, assurant ainsi une répartition équilibrée et efficace de la charge sur les ressources.

L'étude sur le tamis s'est concentrée sur sa capacité à trouver un compromis idéal entre consommation énergétique, due à la puissance de calcul et à la quantité des ressources (cellules) du système, et la qualité de

service (vitesse et quantité des tâches traitées). Pour ce faire, l'étude a été menée dans un environnement à structure de grille infinie : les ressources sont allumées et éteintes selon les besoins. Une ressource sans tâche (grain) est éteinte et s'allume lorsqu'elle reçoit une tâche durant une avalanche. Les résultats ont démontré la capacité du tamis à trouver un compromis quasi-optimal entre efficience énergétique et qualité de service.

Cependant, ces résultats concernent un système infini. Les systèmes de calcul réels, bien qu'ayant des capacités très importantes, ne disposent pas d'un nombre infini de processeurs. Comme nous l'avons vu, le tamis en l'état ne permet pas de gérer une surcharge dans un environnement fini. C'est pourquoi, dans le Chapitre 5, nous explorerons une manière d'améliorer le tamis sur ce point.

Chapitre 4

Robustesse du tas de sable

Table des matières du chapitre

Comme nous l'avons vu dans le chapitre précédent, la SOC, à travers le modèle du tas de sable, est capable de produire un équilibrage de charge efficace. La robustesse est un concept clé pour les systèmes d'équilibrage, définissant leur capacité à faire face à des défaillances inattendues. Il est donc important d'examiner la robustesse du modèle du tas de sable.

Bien que de nombreuses études aient exploré le développement de la SOC au sein de diverses structures (Section 3.3), aucune, à notre connaissance, ne s'est penchée sur son comportement en contexte dégradé, ni sur l'influence des topologies sur l'impact de cette dégradation. Une telle compréhension pourrait offrir des pistes pertinentes pour renforcer la résilience des mécanismes d'équilibrage de charge face aux perturbations ou aux défaillances.

Ce premier travail vise à proposer un modèle à la fois simple et pertinent pour l'analyse de la robustesse dans les systèmes SOC. Notre approche préserve la simplicité du modèle original du tas de sable, tout en y intégrant, de manière cohérente, une diversité de topologies de réseau ainsi que des scénarios de défaillances structurelles.

Pour atteindre cet objectif, nous proposons deux modifications principales du modèle canonique du tas de sable : i) un mécanisme de recâblage permettant de générer un large éventail de topologies de réseau – allant de grilles régulières jusqu'à des structures présentant divers degrés de randomisation et des caractéristiques de petit monde – tout en maintenant un degré de nœud constant, et ii) une approche de simulation progressive de la dégradation, dans laquelle la dynamique du tas de sable est étudiée à partir d'un système pleinement fonctionnel, puis soumise à une augmentation graduelle des défaillances de nœuds, jusqu'à ce que seuls 10% d'entre eux restent opérationnels.

En fin de compte, notre étude vise à explorer les interactions entre la criticalité auto-organisée et les défaillances structurelles au sein des réseaux complexes à travers le prisme du modèle du tas de sable. En examinant différentes structures de réseau (réseau en grille régulière, réseaux de petit monde, réseaux aléatoires) et différents scénarios de défaillance, nous cherchons à identifier des motifs et des principes qui peuvent améliorer la conception de réseaux plus robustes et résilients.

4.1 Cadre d'étude de la robustesse structurelle

En s'appuyant sur les enseignements des études antérieures sur la criticalité auto-organisée et les topologies de réseaux, cette section présente le cadre utilisé pour analyser la robustesse structurelle du modèle du tas de sable Bak-Tang-Wiesenfeld. Si les recherches existantes ont largement étudié l'influence des structures de réseau sur le comportement SOC, la manière dont ces systèmes réagissent à une dégradation progressive de leur architecture demeure une question encore pas ou peu explorée. Pour cela, nous introduisons deux mécanismes clés de modification de la topologie du système : un processus de recâblage, qui nous permet d'explorer les effets de la modification de la connectivité tout en maintenant un degré de nœud fixe, et un processus de dégradation, qui supprime systématiquement des nœuds pour simuler des défaillances structurelles. Ces modifications offrent un environnement contrôlé pour étudier l'interaction entre la criticalité auto-organisée et la robustesse du réseau.

4.1.1 Algorithme de recâblage

La modification des topologies de réseau dans les modèles de SOC implique souvent des changements structurels qui altèrent des propriétés fondamentales telles que la régularité, la directionnalité des connexions ou les mécanismes de dissipation. De nombreuses approches existantes introduisent de telles modifications, rendant les modèles résultants non conservatifs et s'écartant du modèle de tas de sable canonique. Pour préserver le cadre original tout en permettant des altérations contrôlées, nous proposons un algorithme de recâblage qui maintient le degré des nœuds constant tout en introduisant progressivement de l'aléatoire dans la topologie

du réseau.

L'algorithme proposé, inspiré du principe de permutation de G. A. Croes (CROES, 1958), permet la transition d'une topologie de grille régulière à une structure de type petit-monde (WATTS & STROGATZ, 1998), puis à un réseau complètement aléatoire. L'algorithme prend en entrée le graphe et le nombre d'arêtes à recâbler. En augmentant le nombre d'arêtes recâblées, nous ajustons systématiquement la structure du réseau sans modifier les règles fondamentales régissant la dynamique du tas de sable, garantissant ainsi que tout changement observé dans le comportement du système découle directement des altérations topologiques.

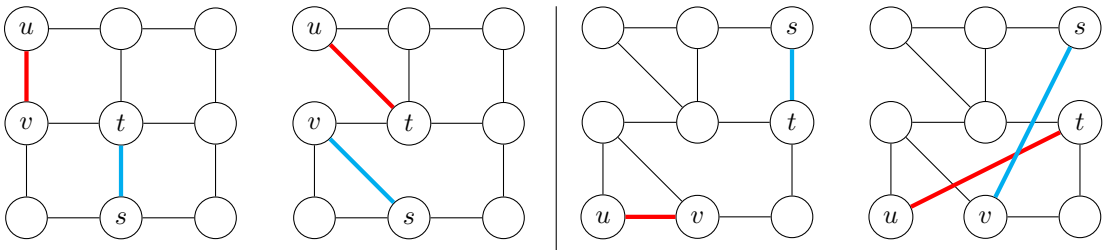
Algorithm 3: Recâblage aléatoire d'arêtes

Input: G : Graphe sur lequel appliquer le recâblage
m : Nombre d'arêtes à recâbler

while $m > 1$ **do**
  Sélection aléatoire de deux arêtes non-recâblées et ne partageant pas de nœud $\{u,v\}$ et $\{s,t\}$ de G
  Suppression des arêtes sélectionnées de G
  Ajout des nouvelles arêtes $\{u,t\}$ et $\{s,v\}$ à G
  $m \leftarrow m - 2$
end

Soit $G = (V, E)$ un graphe non orienté, où V est l'ensemble des nœuds (sommets) et E est l'ensemble des arêtes. Chaque arête $e \in E$ est définie comme une paire non ordonnée de nœuds $\{v_i, v_j\}$ avec $v_i, v_j \in V$.

L'algorithme 3 fonctionne comme suit : deux arêtes, $\{u,v\}$ et $\{s,t\}$, sont sélectionnées aléatoirement dans E, où $u, v, s, t \in V$ sont distincts. Comme illustré dans la Figure 4.1, l'algorithme échange ensuite un nœud de chaque arête, formant ainsi deux nouvelles arêtes, $\{u,t\}$ et $\{s,v\}$. Ce processus est répété $\frac{m}{2}$ fois, où m est le nombre total d'arêtes à reconnecter, garantissant que chaque arête est modifiée une seule fois. De plus, l'algorithme impose une contrainte empêchant la sélection d'arêtes partageant un nœud commun, préservant ainsi l'intégrité structurelle du réseau.



(a) Étape 1 : Sélection des arêtes à recâbler

(b) Étape 1 : Permutation des arêtes sélectionnées

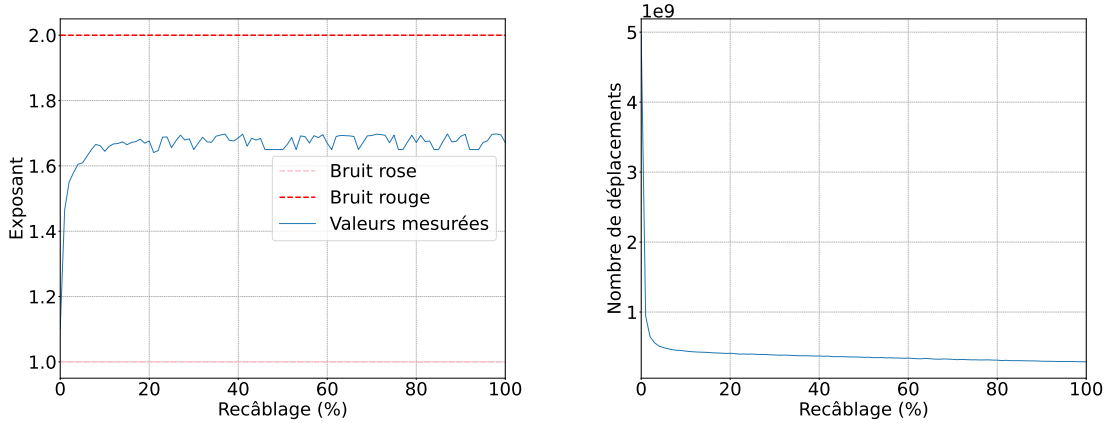
(c) Étape 2 : Sélection d'arêtes additionnelles

(d) Étape 2 : Permutation des arêtes additionnelles

FIGURE 4.1 – Exemple de processus de recâble d'une grille de taille 3 avec $m = 4$.

Les résultats obtenus avec cette méthode de recâblage concordent avec les études précédentes sur les topologies de réseau SOC. Comme le montre la Figure 4.2a, l'exposant de la loi de puissance de la distribution des durées d'avalanche augmente avec le nombre de reconnexions, reflétant un changement dans la dynamique des avalanches. Notamment, les changements les plus significatifs se produisent pour de faibles coefficients de reconnexion, ce qui suggère que même des altérations topologiques minimes peuvent influencer la dynamique

du tas de sable.



(a) Évolution de l'exposant de la loi de puissance pour la distribution des durées d'avalanche.

(b) Évolution de la quantité de mouvement des grains dans le système.

FIGURE 4.2 – Effets du recâblage d'une grille de taille 128 sur la dynamique du tas de sable.

Malgré le maintien d'un degré de nœud constant, la méthode de reconnexion proposée produit des résultats similaires à ceux observés dans la littérature pour d'autres approches de reconnexion qui modifient le degré des nœuds. Plus précisément, l'exposant de la loi de puissance augmente à mesure que la topologie passe d'une structure régulière à un réseau de type petit-monde, puis se stabilise lorsque la topologie de type petit-monde évolue vers un réseau aléatoire.

Une augmentation de l'exposant de la loi de puissance se traduit par une occurrence plus élevée de petites avalanches et une fréquence plus faible de grandes avalanches catastrophiques. Dans ce contexte, un exposant de 1 correspond au *bruit rose*, tandis qu'un exposant de 2 est associé au *bruit rouge*. Selon les travaux fondateurs de Bak, Tang et Wiesenfeld (BAK et al., 1987), une grille régulière produit du bruit rose, tandis qu'une faible quantité de recâblage décale la dynamique des avalanches vers un *bruit rosé*, un état intermédiaire entre le bruit rose et le bruit rouge.

D'un point de vue performance énergétique, l'évolution des avalanches montre qu'un recâblage minime permet de réduire le nombre de sauts effectués par les grains, et par conséquent, la consommation énergétique due aux communications. La Figure 4.2b illustre qu'au début du recâblage (5%), le nombre de grains déplacés par les avalanches est divisé par 10 , atteignant une efficacité jusqu'à 17 fois supérieure lors d'un recâblage complet. Cela démontre une capacité accrue du système à s'auto-réguler comme le suggère l'évolution de l'exposant discuté précédemment.

Ces améliorations notables sont dues à la rapide diminution des distances dans la structure, caractéristique des réseaux de type petit-monde. La Figure 4.3 montre une diminution abrupte de la distance séparant les nœuds d'un bord du système. Cette réduction est particulièrement remarquable pour les nœuds les plus éloignés (courbe orange pointillée). Après les premiers pourcentages de recâblage, les distances moyennes et maximales deviennent très proches et faibles. Cela signifie que, quel que soit le nœud sur lequel se trouve un grain, il n'est qu'à quelques sauts d'être expulsé, optimisant ainsi le processus d'équilibrage de charge.

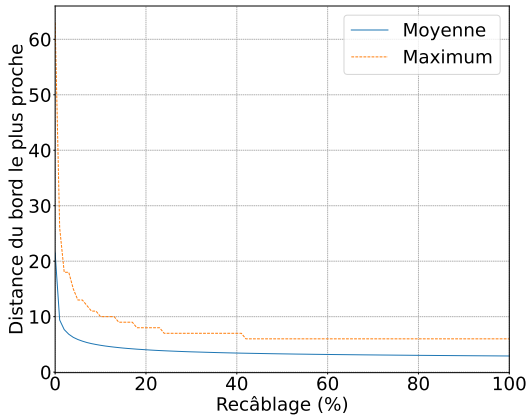


FIGURE 4.3 – Effet du recâblage d'une grille de taille 128 sur la distance séparant les nœuds d'un bord. La première courbe (pleine bleue) correspond à la moyenne des distances, tandis que la deuxième courbe (pointillée orange) présente les distances des nœuds les plus éloignées.

En ce qui concerne la densité de grains représentée dans la Figure 4.4, le processus de reconnexion entraîne seulement une légère diminution. Une densité de 0 indiquerait un tas de sable qui ne retient aucun grain dans le système, tandis qu'une densité de 1 correspond à un système où chaque nœud conserve constamment quatre grains. La faible réduction de densité observée avec la reconnexion suggère que, bien que la structure du réseau change, la capacité globale du système à maintenir une tension nécessaire à son auto-organisation reste pratiquement inchangée.

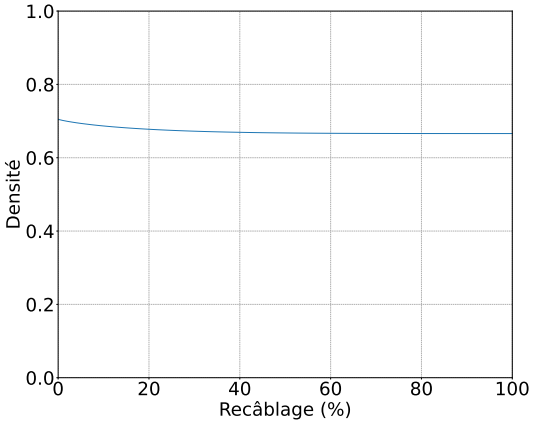


FIGURE 4.4 – Évolution de la densité de grains dans un tas de sable de taille 128 au fil du recâblage.

4.1.2 Processus de dégradation

La structure supportant le modèle de tas de sable est progressivement dégradée pour analyser sa robustesse. Ce processus de dégradation se compose de deux étapes principales. Tout d'abord, une partie des nœuds est retirée aléatoirement du graphe, réduisant ainsi la connectivité globale. Dans une deuxième étape, les zones

isolées qui émergent en raison de la suppression des nœuds sont identifiées et éliminées pour maintenir une structure fonctionnelle. Ce processus est détaillé dans l'Algorithme 4.

Algorithm 4: Processus de dégradation

Input: G : Graphe initial
n : Nombre de nœuds à supprimer
Output: Graphe modifié après dégradation

```
Étape 1 : Supprimer aléatoirement n nœuds de G
Étape 2 : Identifier et supprimer les clusters fermés
for chaque cluster dans G do
    if cluster n'a pas de nœuds bordure then
        Supprimer le cluster de G
    end
end
```

Étape 1 : Suppression des nœuds Un pourcentage défini de nœuds est retiré du système, calculé comme une fraction de la taille totale du graphe. Cette suppression est effectuée aléatoirement tout en garantissant que la structure résultante reste un cadre viable pour la dynamique du tas de sable. À mesure que les nœuds sont supprimés, le graphe se divise naturellement en plusieurs composants séparés, appelés *clusters*.

Étape 2 : Suppression des clusters fermés Après la première étape, certains clusters peuvent devenir complètement isolés, ce qui signifie qu'ils n'ont pas de nœuds frontaliers. Cette situation pose problème pour la dynamique du tas de sable, car les grains piégés à l'intérieur de tels clusters circuleraient indéfiniment sans pouvoir sortir du système. Pour éviter ce problème, les clusters qui ne contiennent pas au moins un nœud frontalier, appelés *clusters fermés*, sont retirés du graphe. En revanche, les clusters ayant au moins un nœud frontalier, appelés *clusters ouverts*, sont conservés, car ils permettent la dissipation des grains.

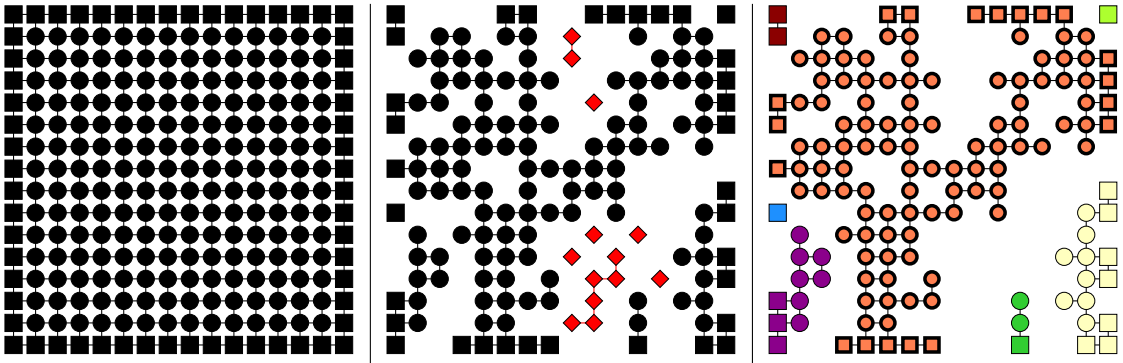
La Figure 4.5 illustre le processus de dégradation en deux étapes, qui commence par un graphe de base initial et applique les deux étapes de dégradation : d'abord, des nœuds aléatoires sont supprimés, puis les clusters fermés sont retirés. Pour démontrer l'impact non linéaire de la deuxième étape sur la structure du graphe, la Figure 4.6 présente plusieurs scénarios avec différents pourcentages de suppression de nœuds. À mesure que la suppression de nœuds augmente, la deuxième étape élimine progressivement les clusters fermés, affectant ainsi de manière significative la structure.

4.1.3 Cadre global : construction de graphe avec recâblage et dégradation

Le cadre global pour l'analyse de la robustesse structurelle du modèle du tas de sable comprend deux étapes séquentielles. Tout d'abord, une grille initiale est générée. Optionnellement, la grille subit un processus de recâblage qui modifie la connectivité tout en préservant un degré constant des nœuds. Ensuite, le graphe recâblé est modifié davantage en appliquant le processus de dégradation décrit précédemment.

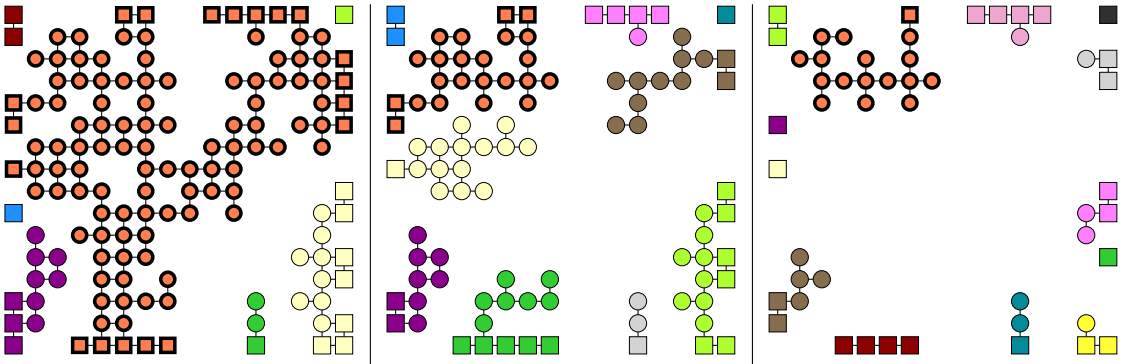
En détail, le cadre fonctionne comme suit :

1. **Génération de la grille :** une grille de taille $S \times S$ est créée pour servir de structure fondamentale.



(a) Grille initiale (b) Dégradation étape 1 (c) Dégradation étape 2

FIGURE 4.5 – Exemple du processus de dégradation sur une grille de 16×16. (a) Structure initiale de la grille, les carrés représentent les nœuds frontaliers et les cercles représentent les nœuds internes. (b) Première étape de dégradation, 40% des nœuds sont supprimés, laissant des nœuds déconnectés d'une bordure (losanges rouges). (c) Deuxième étape de dégradation, les clusters fermés sont supprimés, laissant la structure restante divisée en plusieurs clusters, chacun marqué par une couleur. Le plus grand cluster est mis en évidence avec des contours en gras.



(a) 40% des nœuds supprimés (b) 50% des nœuds supprimés (c) 60% des nœuds supprimés

FIGURE 4.6 – Exemple de trois scénarios de dégradation sur une grille de 16×16, illustrant l'impact non linéaire de la deuxième étape de dégradation. La figure met en évidence la manière dont la suppression des clusters isolés devient plus significative à mesure que le pourcentage de nœuds supprimés augmente.

2. **Reconnexion :** une fraction spécifiée des arêtes est recâblée, introduisant de l'aléatoire dans la topologie tout en maintenant un degré constant pour chaque nœud.

3. **Dégradation :** un pourcentage donné de nœuds est retiré du graphe reconnecté. Ensuite, les clusters fermés (c'est-à-dire ceux qui manquent de nœuds bordures) sont supprimés, garantissant que la structure restante supporte correctement la dynamique du tas de sable.

4. **Ajustement du seuil :** enfin, chaque nœud met à jour son seuil critique en fonction de son nombre actuel de voisins (avec des ajustements supplémentaires pour les nœuds de bordure et d'angle) afin de garantir la compatibilité avec le modèle du tas de sable.

4.2 Dispositif expérimental

Cette section présente la conception expérimentale utilisée pour évaluer le modèle du tas de sable et sa robustesse structurelle. Dans notre cadre, des simulations sont réalisées sur des grilles qui peuvent subir un recâblage optionnel suivi d'une dégradation. Les sous-sections suivantes décrivent les paramètres de simulation et les métriques utilisées pour évaluer à la fois le comportement dynamique des avalanches et le degré de dégradation structurelle.

4.2.1 Paramètres des simulations

Les expériences sont menées sur une grille de taille $S = 128 \times 128$. Chaque simulation s'exécute sur un total de 400 000 cycles, précédés d'une phase d'initialisation de 40 000 cycles pour permettre au système de se stabiliser. À chaque cycle, un grain est ajouté aléatoirement, ce qui peut déclencher une avalanche lorsqu'un nœud dépasse son seuil.

Les principaux paramètres de simulation incluent :

— **Taux de recâblage (Rewiring Rate, RR) :** la fraction des arêtes qui sont reconnectées, variant de 0% à 100% par incréments de 1%. Ce paramètre permet l'étude de la dynamique du tas de sable à travers différentes topologies de réseau, allant d'une grille régulière à des structures de type petit-monde et aléatoires.

— **Taux de suppression des nœuds (Nodes Removal Rate, NRR) :** la fraction de nœuds supprimés pour simuler la dégradation, variant de 0% à 90% par incréments de 1%.

— **Moyennage :** pour chaque combinaison de taux de reconnexion et de suppression des nœuds, les résultats sont moyennés sur 25 simulations indépendantes, chacune utilisant une graine aléatoire différente.

Le Tableau 4.1 résume les paramètres de simulation.

Nom	*Abréviation*	*Valeur*	*Définition*
Grid size	S	128	Taille initiale de la grille
Rewiring rate	RR	0-100%	Fraction des arêtes recâblées
Node removal rate	NRR	0-90%	Taux auquel les nœuds sont initialement supprimés lors de la première étape de la dégradation
Simulation cycles	cycles	400k	Nombre total de cycles par simulation
Random seeds	seeds	25	Nombre de simulations indépendantes moyennées pour chaque combinaison de taux de reconnexion et de suppression des nœuds

TABLE 4.1 – Paramètres des simulations.

4.2.2 Outils d'analyse

Pour évaluer à la fois la performance dynamique du modèle du tas de sable et la robustesse de sa structure de réseau, nous utilisons les métriques suivantes :

— **Densité des grains :** le rapport entre le nombre total de grains et la capacité totale des nœuds. Une

densité de 0 implique une absence de rétention de grains, tandis qu'une densité de 1 indique que chaque nœud est à sa capacité maximale, autrement dit au bord de l'éboulement.

— **Mouvements des grains :** le nombre total de transferts de grains (entre nœuds ou hors du système) pendant les avalanches, servant d'indicateur pour l'efficacité de stabilisation du système, ainsi que pour la consommation énergétique liée à la communication des grains.

— **Taille du graphe :** le nombre de nœuds restants après la dégradation.

— **Clusters ouverts :** le nombre de clusters qui restent viables (c'est-à-dire qui contiennent au moins un nœud frontalier) après le processus complet de dégradation.

— **Ratio du cluster géant par rapport au graphe :** le ratio du nombre de nœuds formant la plus grande composante connexe du graphe dégradé par rapport au nombre total de nœuds restant. Plus le ratio est faible, moins le cluster géant est significatif dans le graphe.

— **Distance des nœuds d'une bordure :** la distance minimale moyenne séparant les nœuds d'un nœud bordure du système.

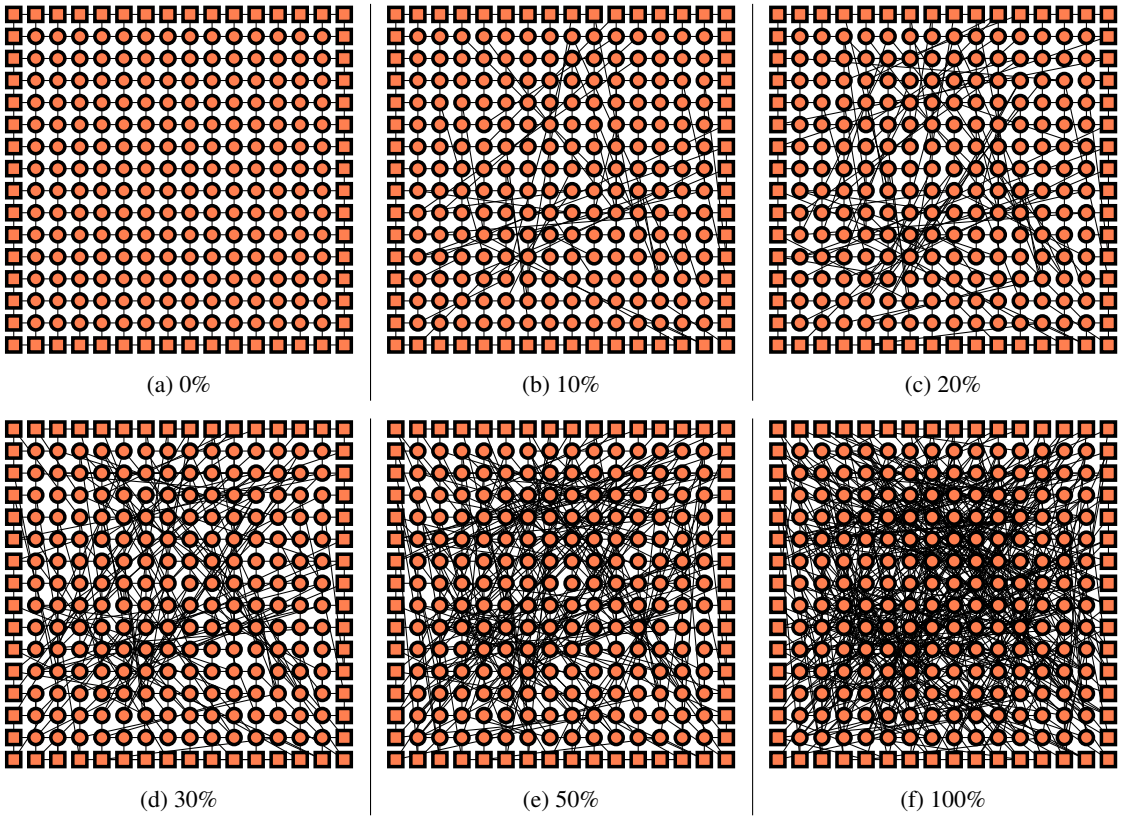
4.3 Étude illustrative

Maintenant que le cadre d'étude est posé, nous allons nous intéresser à quelques cas pratiques illustrant sa mise en œuvre. L'objectif est de comprendre comment la structure générée évolue au cours de son recâblage et de sa dégradation, et comment cela peut impacter la dynamique du tas de sable. Les résultats numériques seront présentés plus tard en Section 4.4. La structure que nous étudierons ici est une grille de taille 16. Nous commencerons par examiner l'impact du recâblage seul, puis nous analyserons la dégradation seule. Enfin, nous explorerons l'effet combiné des deux processus.

4.3.1 Recâblage

La Figure 4.7 illustre l'évolution des connexions dans la grille au fur et à mesure du recâblage. On observe qu'avec un recâblage minime, produisant une structure de type petit-monde, de nombreux raccourcis apparaissent déjà. Ces raccourcis permettent de connecter des régions éloignées, réduisant ainsi considérablement les distances. Au-delà de 30% de recâblage, le mélange des connexions est tel que la structure peut être considérée comme aléatoire.

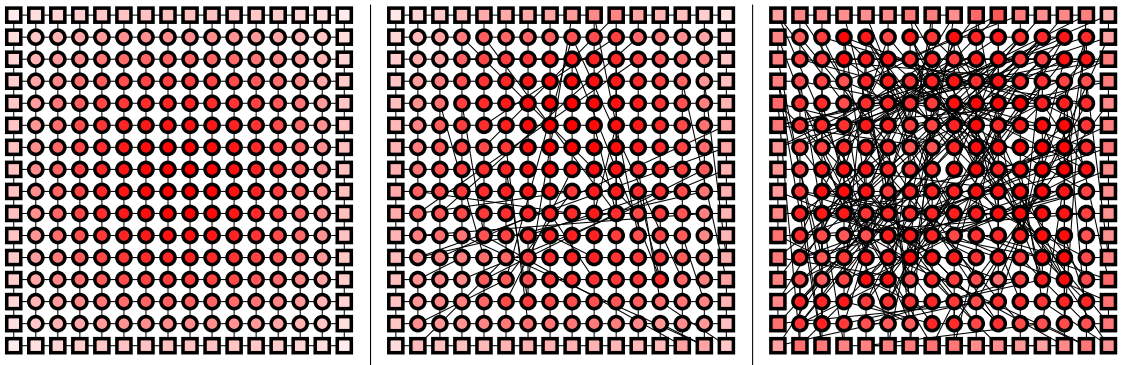
Comme discuté en Section 4.1.1, le recâblage influence la dynamique des avalanches dans le modèle du tas de sable. La Figure 4.8 propose une autre représentation de cette influence : le mouvement des grains sur les nœuds. Plus un nœud est emprunté par des grains au cours des avalanches, plus il est coloré en rouge. Cette représentation montre que, à mesure que le recâblage augmente, les nœuds sont utilisés de manière plus homogène. En revanche, lorsque la structure est dépourvue de raccourcis, le mouvement est principalement concentré en son centre.



(a) 0% (b) 10% (c) 20%

(d) 30% (e) 50% (f) 100%

FIGURE 4.7 – Évolution des connexions dans une grille de taille 16 après recâblage pour différents taux.



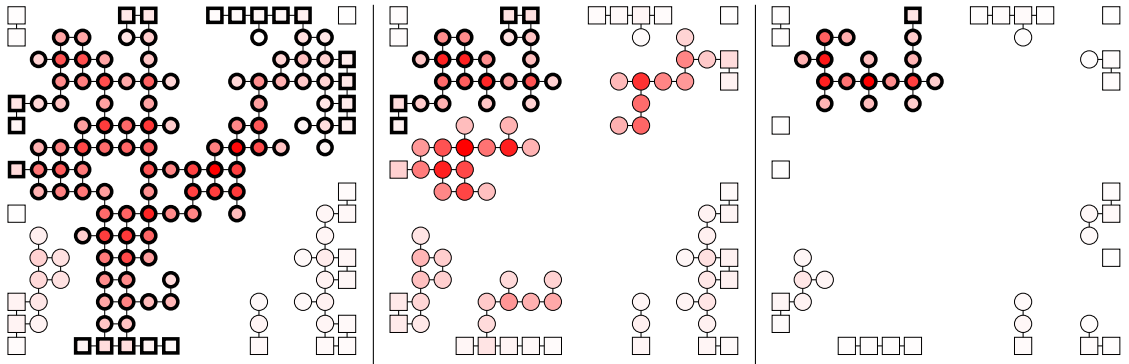
(a) Aucun recâblage (b) 10% de recâblage (c) 50% de recâblage

FIGURE 4.8 – Carte de chaleur du mouvement des grains dans une grille de taille 16, avec et sans recâblage. Plus un nœud est emprunté par des grains au cours des avalanches, plus il est coloré en rouge. L'échelle est indépendante d'une représentation à l'autre. Cela permet de voir que le recâblage homogénéise le mouvement des grains.

4.3.2 Dégradation

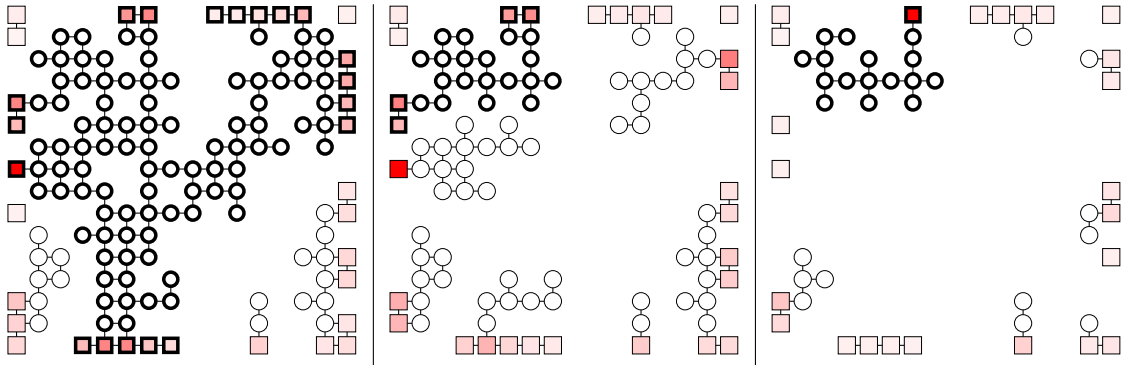
Pour ce qui est du processus de dégradation, son impact sur la structure a été introduit en Section 4.1.2 et illustré par les Figures 4.5 et 4.6. De manière similaire à l'analyse du recâblage, nous pouvons examiner comment la dégradation influence l'utilisation des nœuds. La Figure 4.9 montre cette influence pour des grilles où 40%, 50%, et 60% des nœuds ont été initialement supprimés. On observe que, plus un nœud est éloigné d'une bordure, plus il est emprunté par les grains, comme c'est le cas pour la grille initiale (Figure 4.8). Cependant,

à mesure que la dégradation progresse, les plus gros clusters disposent de moins en moins de nœuds bordure pour éjecter les grains, favorisant ainsi des avalanches relativement longues par rapport à leur taille. De plus, ces clusters concentrent la majorité des grains en raison de leur taille. C'est pourquoi, comme le montre la Figure 4.10, les rares nœuds bordure des plus gros clusters éjectent beaucoup plus de grains du système que les autres.



(a) 40% des nœuds supprimés (b) 50% des nœuds supprimés (c) 60% des nœuds supprimés

FIGURE 4.9 – Carte de chaleur du mouvement des grains dans une grille de taille 16 après dégradation. Plus un nœud est emprunté par des grains au cours des avalanches, plus il est coloré en rouge. L'échelle est indépendante d'une représentation à l'autre. L'éclatement de la structure en de multiples petits morceaux diminue le nombre de nœuds de bordure pour les clusters les plus gros, concentrant alors la majorité du mouvement.

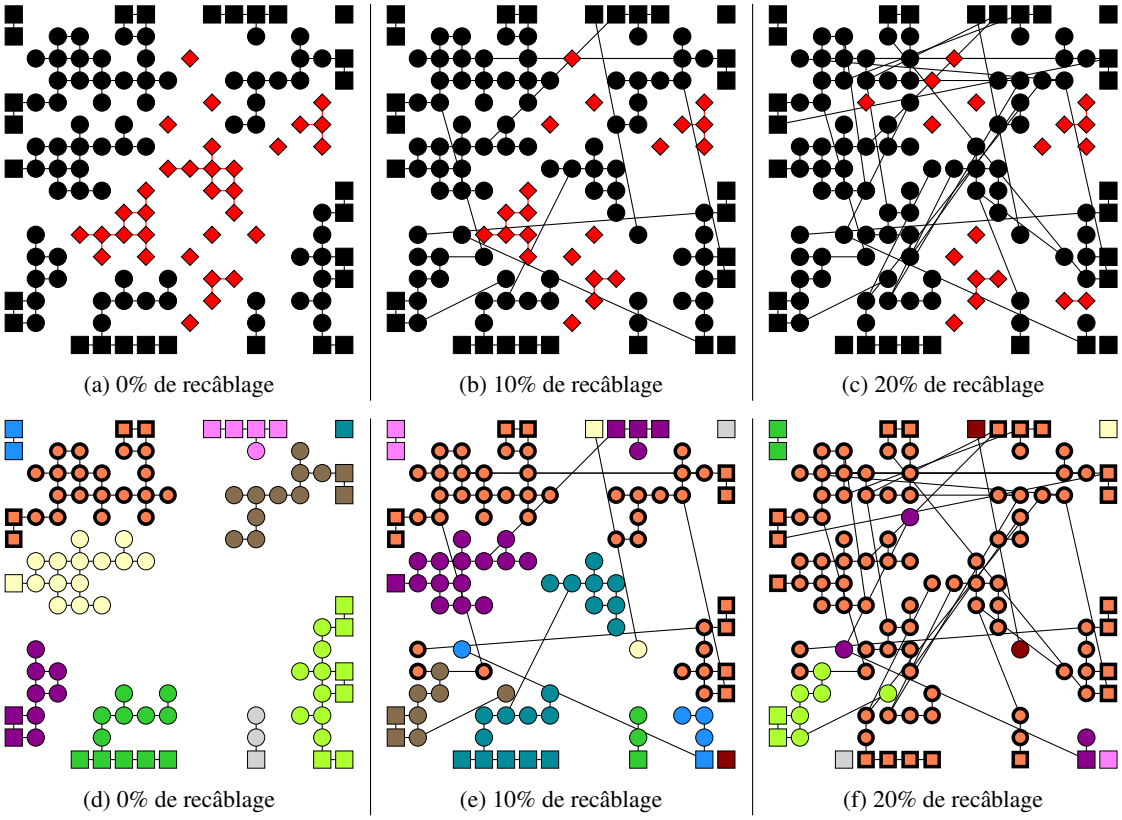


(a) 40% des nœuds supprimés (b) 50% des nœuds supprimés (c) 60% des nœuds supprimés

FIGURE 4.10 – Carte de chaleur de l'éjection des grains pour une grille de taille 16 après dégradation. Plus un nœud éjecte des grains au cours des avalanches, plus il est coloré en rouge. L'échelle est indépendante d'une représentation à l'autre.

4.3.3 Recâblage et dégradation

Maintenant que nous avons étudié l'impact des deux processus séparément, nous pouvons nous intéresser à leur combinaison. La Figure 4.11 illustre l'évolution d'une grille dont 50% des nœuds sont initialement supprimés. On observe que le nombre de nœuds supprimés lors de la deuxième étape du processus de dégradation diminue presque de moitié avec seulement 20% de recâblage. De plus, un cluster (réellement) géant, essentiel à la bonne auto-régulation du système à grande échelle, apparaît. En revanche, lorsqu'il n'y a pas de recâblage, aucun cluster ne se démarque réellement, tous étant d'une taille très modeste.



(a) 0% de recâblage (b) 10% de recâblage (c) 20% de recâblage

(d) 0% de recâblage (e) 10% de recâblage (f) 20% de recâblage

FIGURE 4.11 – Évolution de la structure d'une grille de taille 16 dont 50% des nœuds sont supprimés, sans recâble puis avec un recâblage de 10 et 20%. La première ligne illustre l'évolution du nombre de clusters fermés (identifiés par des losanges rouges pour leurs nœuds), tandis que la seconde montre les clusters restants une fois le processus de dégradation complet.

Ces processus de recâblage et de dégradation offrent un cadre d'étude puissant pour analyser comment la SOC, à travers le modèle du tas de sable, se comporte dans une multitude de structures. Maintenant que nous avons les clés de compréhension de ces processus et de leurs impacts, nous pouvons analyser comment le modèle du tas de sable et sa structure y réagissent.

4.4 Analyse des résultats

Cette section propose une analyse complète des résultats des simulations. Les expériences évaluent deux aspects clés : la robustesse structurelle de diverses topologies de réseau face à la dégradation et l'évolution de la dynamique du tas de sable en réponse à ces changements structurels. Nous commencerons par examiner la manière dont les différentes structures de réseau introduisant du recâblage se détériorent, puis nous poursuivrons avec le comportement dynamique du modèle du tas de sable dans ces structures. La discussion finale résume les principales conclusions et leurs implications.

4.4.1 Robustesse des différentes structures

La robustesse du réseau, telle que définie par DEKKER et COLBERT (2004), fait référence au nombre minimum de suppressions de nœuds nécessaires pour déconnecter un réseau. Dans notre cadre d'étude du tas de sable, le réseau se dégrade progressivement jusqu'à s'effondrer complètement. Les grilles sont particulièrement vulnérables aux défaillances structurelles. Notre objectif est de maximiser le nombre de nœuds conservés dans un seul cluster connecté pendant cette dégradation, retardant ainsi le début d'une fragmentation rapide et étendue. Cette approche améliore la tolérance aux pannes du réseau, non pas en empêchant les premières scissions, qui surviennent tôt, mais en retardant la rupture critique en de nombreux petits clusters. À cette fin, nous appliquons la technique de recâblage, détaillée dans la Section 4.1.1, pour renforcer la robustesse face à la dégradation.

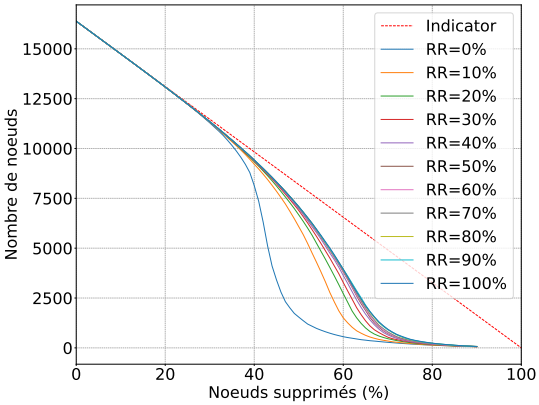
La Figure 4.12 illustre l'évolution structurelle pour différents taux de recâblage pendant la dégradation. Pour la grille régulière (RR=0%), un seuil critique apparaît entre 30% et 40% de suppressions de nœuds, au-delà duquel commence l'effondrement. Ce point est marqué par un écart abrupt de la taille du graphe par rapport à l'indicateur de dégradation linéaire (ligne pointillée rouge dans la Figure 4.12a). Le réseau se fragmente alors en de nombreux petits clusters, dont plusieurs manquent de nœuds frontaliers et sont ensuite supprimés (Figure 4.12b). Simultanément, la dominance de la composante géante diminue (Figure 4.12c), reflétant la désintégration en composantes plus petites et moins significatives.

De manière remarquable, même un recâblage minimal améliore considérablement la robustesse. Le seuil d'effondrement est repoussé de 10 à 20%, permettant finalement au réseau de supporter jusqu'à 60% de suppressions de nœuds avant une fragmentation critique. Cette amélioration découle de la perturbation des métriques de distance originales du réseau par le recâblage, raccourcissant les chemins, en particulier vers les nœuds bordures (Figure 4.12d), et améliorant ainsi la connectivité et la stabilité face à la dégradation.

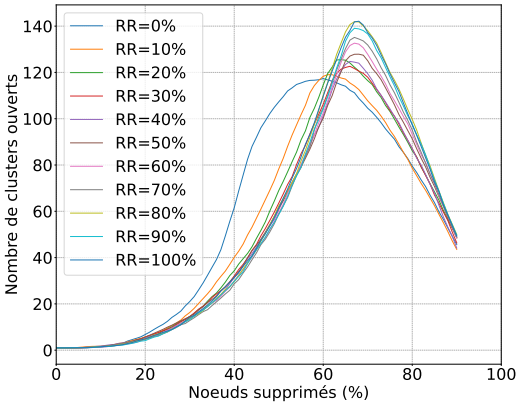
4.4.2 Évolution de la dynamique du tas de sable

Les changements dans la topologie du réseau modifient naturellement le comportement du modèle du tas de sable canonique, affectant particulièrement sa capacité à maintenir la criticalité auto-organisée à mesure que la structure subit une dégradation. Cette sous-section examine comment la dynamique du modèle évolue avec des suppressions progressives de nœuds pour différents taux de recâblage, en se concentrant sur des métriques clés telles que la densité de grains et le nombre total de mouvements des grains. En analysant ces métriques, il devient évident que la reconnexion non seulement renforce la robustesse structurelle, mais façonne également la performance du modèle sous contrainte. La Figure 4.13 illustre cette progression : la Figure 4.13a met en évidence les changements de densité de grains, et la Figure 4.13b illustre le nombre total de mouvements de grains pour chaque configuration de réseau.

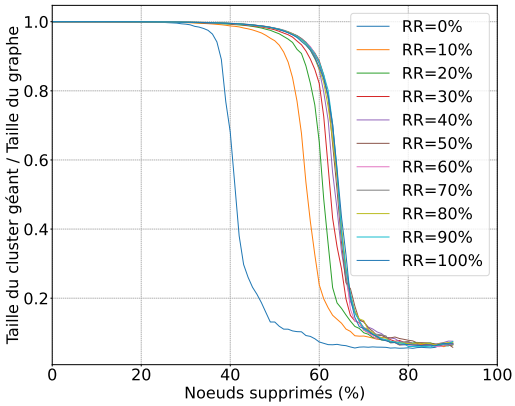
Dans une grille sans recâblage (RR = 0%), le début de la dégradation déclenche une réponse en deux phases. À mesure que la proportion de nœuds supprimés approche environ 40%, les chemins entre les nœuds et les points de sortie deviennent plus longs, augmentant le temps que les grains passent dans le système. Ce parcours prolongé entraîne une augmentation des mouvements de grains (Figure 4.13b), atteignant jusqu'à 51



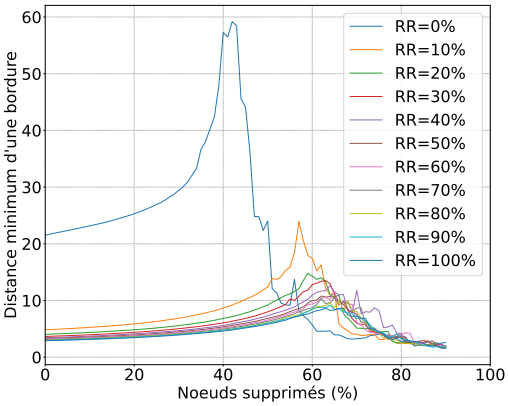
(a) Taille du graphe.



(b) Nombre de clusters ouverts.

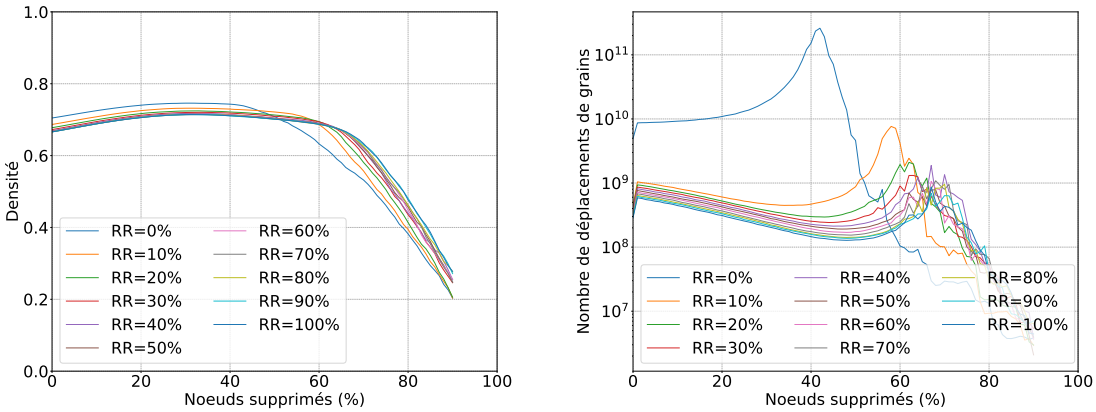


(c) Ratio du cluster géant par rapport au graphe.



(d) Distance moyenne des nœuds d'une bordure.

FIGURE 4.12 – Évolution de l'impact de la dégradation sur différentes structures avec un taux de recâblage allant de 0 à 100% par incréments de 10%.



(a) Densité des grains.

(b) Mouvements des grains.

FIGURE 4.13 – Évolution de l'impact de la dégradation sur la dynamique du tas de sable pour des structures avec un taux de recâblage allant de 0 à 100% par incréments de 10%.

fois la valeur de base à environ 42% de suppression de nœuds, accompagnée d'une légère augmentation de la densité de grains (Figure 4.13a). Cependant, une fois que le réseau dépasse ce seuil, la fragmentation en clusters ouverts plus petits réduit brusquement les longueurs des chemins, entraînant une diminution des mouvements de grains et de la densité. À ce stade, le système perd la tension nécessaire pour maintenir la SOC, et les avalanches à grande échelle ne se produisent plus.

L'introduction du recâblage modère ces effets, retardant le point auquel le réseau subit une fragmentation brutale. Des routes plus courtes vers les points de sortie distribuent les grains de manière plus uniforme pendant une période plus longue, maintenant des niveaux plus élevés de mouvements de grains (Figure 4.13b) et de densité (Figure 4.13a) plus en avant dans le processus de dégradation. La connectivité améliorée augmente le seuil d'effondrement d'au moins 15% de suppressions de nœuds supplémentaires par rapport à la grille sans recâble. Les taux de recâblage intermédiaires (par exemple, de 10% à 90%) produisent un gradient de résultats, avec des fluctuations plus prononcées de la densité et des mouvements à mesure que le taux de recâblage augmente. Globalement, le recâblage s'avère bénéfique pour préserver la SOC et ralentir la transition vers un état fragmenté et sous-critique.

4.4.3 Discussion

Cette étude fait progresser la compréhension de la robustesse structurelle et de la criticalité auto-organisée dans les réseaux complexes en examinant le modèle du tas de sable de Bak-Tang-Wiesenfeld à travers diverses topologies et scénarios de dégradation. Notre analyse révèle des éléments clés sur la robustesse des réseaux, la dynamique des avalanches et le maintien de l'état critique, avec des implications pour les systèmes technologiques et biologiques.

Robustesse des réseaux : Les résultats montrent que les grilles régulières maintiennent le comportement de SOC jusqu'à un taux de défaillance des nœuds de 30 à 40%, tandis que les réseaux avec un recâblage

minimal (par exemple, 10% de reconnexion) tolèrent jusqu'à 55% de pertes de nœuds avant une fragmentation critique. Cette disparité souligne le rôle crucial de la structure topologique dans l'amélioration de la résilience. Les propriétés de « petit monde » introduites par le recâblage, telles que la réduction des longueurs de chemin et l'amélioration de la connectivité, reflètent des caractéristiques observées dans des systèmes réels résilients. Par exemple, dans les réseaux de processeurs, où des défaillances de nœuds (processeurs) ou de liens (connexions) peuvent survenir en raison de problèmes matériels, l'adoption de telles topologies pourrait retarder la défaillance du système, offrant une stratégie pratique pour concevoir des infrastructures technologiques tolérantes aux pannes, comme les systèmes de calcul distribué ou les réseaux énergétiques.

Dynamique des avalanches : Le recâblage maintient l'état critique pendant une dégradation prolongée en améliorant l'efficacité de la redistribution des grains induite par les avalanches, comme en témoignent une densité de grains plus élevée et des mouvements de grains optimisés (Section 4.4.2). Cette efficacité réduit l'énergie nécessaire à la stabilisation du système. Dans ce contexte, les topologies recâblées minimisent les mouvements de grains inutiles, similaires à la réduction des coûts de communication entre les cœurs ou les nœuds d'un réseau informatique. Un tel comportement suggère des applications potentielles dans la distribution dynamique de la charge de travail, où la réorganisation des connexions pourrait réduire la consommation d'énergie lors des transferts de données, améliorant ainsi la durabilité des systèmes informatiques.

Maintien de l'état critique : La capacité des réseaux reconnectés à préserver la SOC au minimum jusqu'à 55% de défaillances de nœuds met en lumière leur potentiel pour la gestion des pannes dans les systèmes à haute criticité, tels que les dispositifs embarqués ou les réseaux en temps réel. Lorsqu'un nœud tombe en panne, les topologies recâblées facilitent un réacheminement rapide, atténuant la dégradation des performances ou la perte de données. La criticalité prolongée réduit l'énergie nécessaire pour retrouver la stabilité après une perturbation, diminuant ainsi la dépendance aux mécanismes de redondance. Dans les réseaux de processeurs, cela pourrait prolonger la durée de vie des composants et optimiser l'utilisation de l'énergie en évitant des opérations de récupération excessives, en ligne avec les objectifs d'efficacité et de résilience.

Connexions avec le fonctionnement cérébral et les crises épileptiques : Les résultats résonnent avec la SOC dans les systèmes biologiques, notamment les réseaux neuronaux du cerveau. Les avalanches neuronales, une caractéristique de la SOC, permettent un traitement efficace de l'information dans les cerveaux sains (BEGGS & PLENZ, 2003 ; HAHN et al., 2010). Cependant, des perturbations structurelles, analogues aux défaillances de nœuds dans ce modèle, peuvent déstabiliser cet équilibre, poussant le système vers un état supercritique, comme observé lors des crises épileptiques (MEISEL et al., 2012). L'observation que les grilles régulières s'effondrent à 30-40% de perte de nœuds est parallèle à la manière dont une déconnexion neuronale excessive peut déclencher une activité incontrôlée. Inversement, la robustesse accrue des réseaux recâblés suggère que le maintien d'un équilibre topologique, similaire à la modularité hiérarchique du cerveau (S.-J. WANG & ZHOU, 2012), pourrait retarder de telles transitions. Cette analogie implique que les enseignements tirés du modèle du tas de sable pourraient aiguiller vers des stratégies pour stabiliser les réseaux neuronaux,

potentiellement en atténuant les crises épileptiques en préservant une connectivité critique.

Implications Plus Larges : L'effondrement brutal des métriques de SOC au-delà de seuils critiques (par exemple, 55% de perte de nœuds dans les réseaux recâblés) indique que, bien que les modifications topologiques renforcent la résilience, des limites structurelles inhérentes persistent. Cela soulève des questions sur les transitions vers des états chaotiques ou sous-critiques dans les systèmes artificiels et naturels. Par exemple, dans les réseaux électriques, le dépassement d'un seuil de défaillance pourrait conduire à des pannes en cascade (SHENGWEI MEI et al., 2008), tandis que dans les réseaux financiers, cela pourrait précipiter des krachs boursiers (BIONDO et al., 2015). Ces résultats fournissent ainsi un cadre pour anticiper et gérer de tels points de basculement à travers les domaines.

En résumé, cette étude non seulement éclaire l'interaction entre la topologie et la résilience de la SOC, mais établit également un lien entre les perspectives théoriques et les applications pratiques. En soulignant les parallèles avec le fonctionnement cérébral et les systèmes technologiques, elle met en lumière la pertinence universelle de ces principes et ouvre la voie à des avancées interdisciplinaires.

4.5 Conclusion

Cette étude explore l'interaction entre la criticité auto-organisée (SOC) et la robustesse structurelle dans les réseaux complexes, en utilisant le modèle du tas de sable de Bak-Tang-Wiesenfeld comme cadre d'analyse. En examinant les effets de la topologie du réseau et de la dégradation sur le comportement de la SOC, nous découvrons des mécanismes qui améliorent la résilience dans divers systèmes.

Nos principales contributions sont les suivantes :

1. Robustesse structurelle améliorée : les grilles régulières maintiennent la SOC jusqu'à un taux de défaillance des nœuds de 30 à 40%, tandis que les réseaux avec un recâblage minimal (10%) supportent jusqu'à 55% de pertes de nœuds. Cela démontre que de petits ajustements topologiques renforcent considérablement la résilience, offrant un modèle pour la conception de réseaux tolérants aux pannes.

2. Dynamique des avalanches optimisée : le recâblage réduit les longueurs moyennes des chemins pour la redistribution des grains, retardant l'effondrement structurel et maintenant une densité de grains plus élevée pendant la dégradation. Cette efficacité préserve l'état critique, reflétant les comportements adaptatifs dans les systèmes naturels et artificiels.

3. Gains d'efficacité énergétique : la réduction des mouvements de grains dans les réseaux recâblés diminue l'énergie nécessaire à la stabilisation, un principe applicable à la minimisation de l'utilisation des ressources dans les systèmes technologiques comme les réseaux de processeurs ou les réseaux électriques.

4. Pertinence dans le monde réel : les résultats ont des implications pratiques pour les infrastructures critiques, par exemple, l'amélioration de la tolérance aux pannes dans les systèmes informatiques, et les

systèmes biologiques, comme la stabilisation des réseaux neuronaux pour prévenir les crises épileptiques. Cette double applicabilité souligne l'universalité de la résilience de la SOC.

5. Orientations de recherche : l'étude met en lumière le potentiel des réseaux auto-adaptatifs qui se reconnectent dynamiquement en réponse aux pannes, ouvrant la voie à l'exploration future de stratégies de résilience en temps réel.

Ces contributions révèlent que la topologie du réseau est un levier crucial pour maintenir la SOC sous contrainte structurelle. En exploitant un recâblage minimal, nous pouvons concevoir des systèmes capables de supporter des perturbations importantes tout en optimisant l'utilisation de l'énergie, une découverte pertinente pour des domaines tels que la distribution d'énergie, les neurosciences, et dans notre cas l'équilibrage de charge. À l'avenir, l'étude des mécanismes auto-adaptatifs pourrait encore améliorer les modèles de SOC, permettant aux réseaux de maintenir de manière autonome leur robustesse et leur efficacité dans des applications telles que les réseaux intelligents ou les prothèses neurales. Ce travail fournit ainsi une base pour construire des systèmes complexes résilients et durables, capables de prospérer face à l'adversité.

Chapitre 5

Le tamis auto-adaptatif

Table des matières du chapitre

Le modèle de tamis, dans sa version initiale présentée en Section 3.5.3, constitue une modélisation à la fois simple et efficace d'un système de traitement de tâches. Il a démontré sa capacité à équilibrer dynamiquement la charge de travail et à traiter efficacement les tâches qui lui sont confiées. Toutefois, comme évoqué précédemment, **ce modèle a été étudié dans un environnement infini**, ce qui ne reflète pas les contraintes réelles d'un système de traitement, où le nombre de ressources est limité.

Dans ce chapitre, nous introduisons une évolution du modèle : le tamis auto-adaptatif. L'objectif principal de ce nouveau modèle est de rendre le tamis adaptatif, afin qu'il soit capable de fonctionner durablement dans un système contraint en taille, tout en préservant sa décentralisation. Nous montrerons, à travers différentes études, que les mécanismes d'adaptation proposés permettent au système de faire face au travail qui lui est soumis tout en conservant ses propriétés d'auto-organisation, en particulier lorsqu'il est soumis à une surcharge continue.

Nous commencerons par discuter de l'introduction explicite des contraintes de taille dans le modèle, des problématiques qu'elles soulèvent, ainsi que des leviers intrinsèques au modèle que nous mobilisons pour y répondre. Nous nous intéresserons ensuite à une première amélioration portant sur les seuils critiques dynamiques, avant d'aborder la modélisation de l'évolutivité des capacités de tamisage au niveau cellulaire. Nous présenterons et analyserons ensuite deux mécanismes d'adaptation de ces capacités : un modèle basé sur un calcul d'entropie locale, et un modèle fondé sur un protocole de bavardage. Enfin, ces deux approches seront comparées afin d'évaluer leurs performances respectives dans différents contextes de charge.

5.1 Un environnement limité pour le tamis

Afin d'introduire la contrainte de limitation des ressources, l'automate cellulaire n'évolue plus dans une grille classique comme c'était le cas jusqu'à présent, mais dans une topologie toroïdale. Cette configuration suppose que les bords de la grille sont connectés entre eux, formant une surface continue équivalente à celle d'un tore. Ce choix permet d'éliminer les effets de bord en uniformisant le voisinage des cellules, et ainsi de simuler un espace homogène, fini en nombre de cellules mais sans frontières, dans lequel chaque cellule possède exactement le même nombre de voisines.

Si l'environnement du tamis devient limité, la charge que peut gérer le système est bornée par sa capacité de traitement globale. Une charge excédant ce seuil provoquera une avalanche qualifiée d'infinie ; c'est-à-dire que le système sera trop saturé pour trouver une répartition stable de la charge, et l'avalanche se poursuivra indéfiniment, entraînant une dépense énergétique importante et un blocage des simulations. La problématique de la gestion de la surcharge est donc soulevée pour le modèle initial du tamis. Nous explorerons ici deux options intrinsèques au modèle permettant d'y répondre : le seuil critique d'éboulement et la capacité de tamisage des cellules.

Le seuil critique est un paramètre sensible, car il définit la quantité de grains qu'une cellule peut retenir

avant de s'ébouler. Une valeur trop élevée peut complètement annuler le déclenchement des avalanches, et par conséquent, tout mécanisme d'auto-organisation dans le système. Le seuil critique ne peut donc pas être fixé à une valeur élevée à priori. Une approche adaptative semble plus adéquate pour le faire évoluer et ainsi contrôler les avalanches. Cependant, l'adaptation de ce paramètre seul ne ferait que repousser le problème. En effet, si le système n'est pas en mesure d'évacuer le surplus de grains, contrôler les avalanches n'y changera rien et le seuil critique ne fera qu'augmenter au cours du temps. Il est donc nécessaire de s'intéresser également à la capacité de tamisage des cellules.

Une première approche consisterait à augmenter substantiellement la capacité des cellules afin de rendre le tamis capable de traiter n'importe quelle quantité de grains. Toutefois, une telle augmentation arbitraire va à l'encontre de l'objectif d'optimisation de l'efficience énergétique du système. De plus, comme discuté en Section 3.5.3, la capacité de tamisage influence directement la dynamique des avalanches. Une capacité trop élevée induit un traitement très rapide des grains et donc une réduction, voire une absence, de la tension nécessaire à l'auto-organisation du système. Une fois encore, une approche adaptative semble préférable, afin de maintenir la capacité de traitement des cellules à un niveau optimal, permettant de traiter la charge tout en maintenant la tension nécessaire à l'équilibrage.

En définitive, pour rendre le tamis plus robuste et résilient, nous avons choisi la voie de l'adaptation combinée de ses paramètres en fonction de la charge. L'objectif est de produire ce comportement d'auto-adaptation de manière émergente, en complément de l'auto-organisation naturelle apportée par les mécanismes du tas de sable. Nous allons explorer dans les sections suivantes différentes modélisations de cette auto-adaptation.

5.2 Seuil critique dynamique

Le seuil critique dynamique est une stratégie inspirée des travaux de QI et PFENNINGER (2015), visant à contrôler les avalanches en modifiant dynamiquement le seuil d'éboulement des cellules touchées par une avalanche. L'objectif est d'augmenter la rétention des grains par les cellules lors d'avalanches de grande ampleur, afin de ralentir leur propagation. En revanche, lors d'avalanches plus modestes, les seuils tendent à retrouver une valeur plus basse, permettant ainsi la survenue de nouvelles avalanches importantes. Nous proposons ici une version simple, bien qu'efficace, d'adaptation du seuil critique.

Nous commencerons par proposer une modélisation du seuil critique dynamique, en détaillant les principes et mécanismes qui le régissent. Dans un second temps, ce modèle sera mis à l'épreuve au travers de différents scénarios expérimentaux, afin d'illustrer concrètement ses effets sur la dynamique des avalanches et la stabilité du système.

5.2.1 Modélisation

Le mécanisme du seuil critique dynamique s'incorpore à l'intérieur des avalanches. À chaque étape d'une avalanche, une cellule qui reçoit au moins deux grains de ses voisines et devient instable voit son seuil incrémenté de 2. En revanche, si elle ne reçoit qu'un grain et que diminuer son seuil ne la rendra pas instable,

celui-ci est décrémenté de 1, jusqu'à un minimum correspondant au nombre de cellules voisines, soit 4 dans notre étude sur une grille définie par un voisinage de von Neumann.

La valeur d'incrémentation doit être plus élevée que celle de la décrémentation afin que l'augmentation des seuils ne soit pas immédiatement compensée par la baisse, sans quoi le mécanisme serait inefficace. De plus, les conditions annexes au nombre de grains reçus sont nécessaires afin de mieux contrôler la modification du seuil.

Pour le cas de l'incrémentation, si celle-ci se produit dès qu'une cellule reçoit au moins deux grains, les seuils ne feraient qu'augmenter continuellement au fil du temps. La seconde partie de la condition restreint l'augmentation du seuil uniquement lorsque c'est nécessaire, c'est-à-dire lorsqu'il y a un mouvement conséquent et que l'ajout de grains rend la cellule instable. Pour la diminution du seuil, imposer que la cellule ne soit pas instable après la décrémentation permet de conserver un seuil qui évite de poursuivre l'avalanche, et ainsi freiner sa propagation. Sans cette condition, les seuils ne feraient qu'augmenter et diminuer en permanence, relançant des avalanches qui devaient se terminer.

L'Algorithme 5 propose un pseudo-code du seuil critique dynamique : à chaque étape d'une avalanche du tas de sable canonique (Algorithme 1), une étape de mise à jour du seuil est effectuée. Bien que simple, ce mécanisme a des effets à plusieurs niveaux dans le système.

Algorithm 5: Gestion d'une avalanche avec seuil critique dynamique

Input: G : grille

```
1  while au moins une cellule de G est instable do
       /* Éboulement des cellules instables                          */
2      foreach cellule de G do
           /* Éboulement de la cellule                               */
3          if cellule.grains ≥ cellule.seuil then
4              foreach voisine de cellule do
5                  voisine.grains ← voisine.grains + 1
6              end
7              cellule.grains ← cellule.grains - 4
8          end
9      end
10
       /* Mise à jour du seuil                                       */
11     foreach cellule de G do
12         if cellule a reçu au moins 2 grains and cellule est instable then
13             cellule.seuil ← cellule.seuil + 2
14         else if cellule a reçu exactement 1 grain and réduire le seuil ne rend pas cellule instable then
15             cellule.seuil ← max(4, cellule.seuil - 1)
16         end
17     end
18 end
```

D'abord, l'auto-adaptation du seuil critique affecte partiellement le déclenchement des éboulements. L'augmentation du seuil d'une cellule permet de retarder momentanément son instabilité. Par exemple, une cellule disposant de 2 grains et qui en reçoit 2 devrait s'ébouler, mais l'augmentation de son seuil à 6 la maintient

stable. Toutefois, le mécanisme d'augmentation du seuil n'empêche pas complètement les éboulements ; une cellule proche de l'éboulement ($grains = seuil - 1$) s'éboulera tout de même si elle reçoit un, trois ou quatre grains. En outre, la diminution du seuil permet aux cellules de retrouver progressivement et de manière contrôlée leur état initial pour garantir que de nouvelles avalanches puissent survenir à l'avenir.

Ensuite, l'augmentation progressive des seuils permet aux cellules d'absorber de plus en plus de grains lors d'avalanches de grande ampleur, réduisant ainsi leur durée et leur propagation. Ce mécanisme s'avère particulièrement pertinent dans le contexte du tamis en environnement limité, où il est impératif que les avalanches trouvent une issue autre que l'éjection de grains, comme c'est le cas dans le modèle canonique du tas de sable. Grâce à cette auto-adaptation des seuils, le tamis est capable de stocker temporairement une surcharge de grains, avant de retrouver un comportement d'auto-organisation performant dès que les cellules ne sont plus saturées.

Ainsi, le seuil dynamique permet d'éviter des éboulements lorsque le nombre de grains impliqués dans l'avalanche est élevé, amortissant les avalanches de grande ampleur, tout en permettant des éboulements nécessaires à l'auto-organisation. La Figure 5.1 illustre quelques-unes de ces situations. La cellule colorée en jaune devrait normalement s'ébouler deux fois, mais grâce au seuil dynamique, la seconde est évitée.

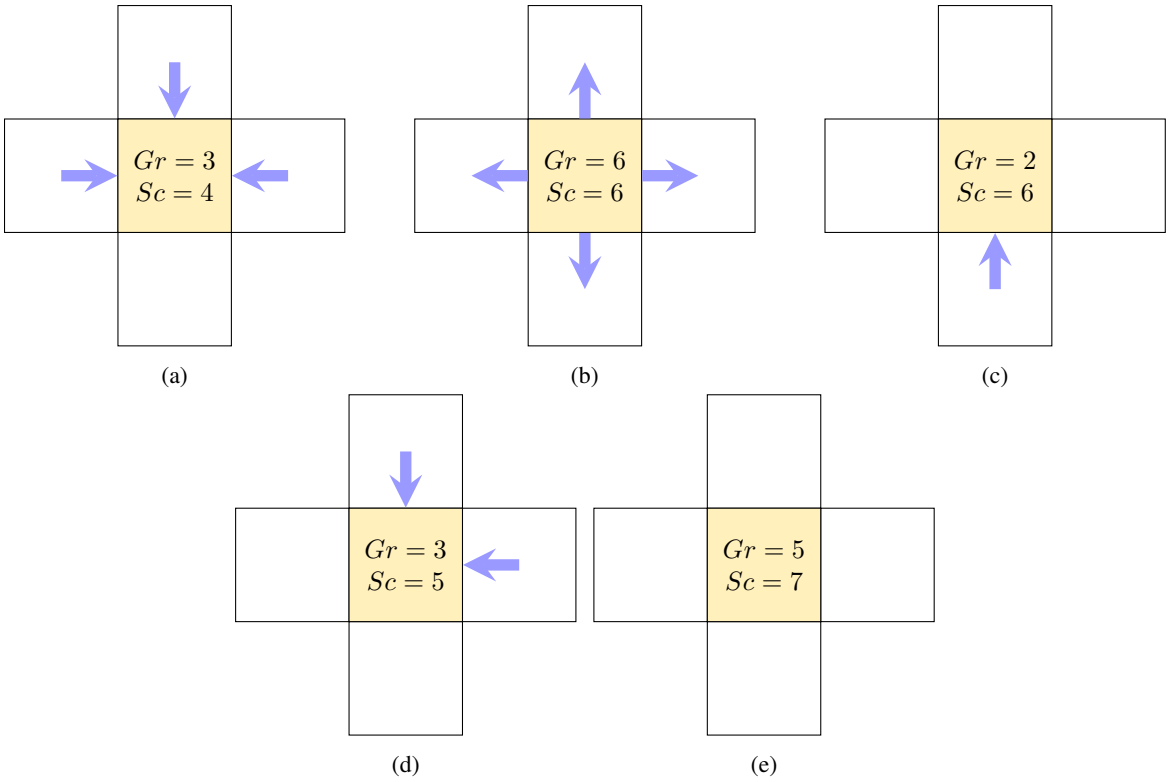


FIGURE 5.1 – Illustration de l'évolution du seuil critique d'éboulement d'une cellule durant une avalanche. Trois grains arrivent sur la cellule jaune (a), augmentant son seuil critique (Sc) et son nombre de grains (Gr) et faisant s'ébouler la cellule (b). L'avalanche se poursuit et un nouveau grain arrive (c), faisant diminuer le seuil (d), puis deux nouveaux grains tombent sur la cellule. Elle finit par se stabiliser grâce au seuil dynamique (e).

5.2.2 Cas d'étude

Nous soumettons ici le mécanisme de seuil critique dynamique à une série de quatre scénarios, afin d'illustrer son impact sur la propagation des avalanches. Chaque scénario consiste en l'observation d'une avalanche résultant d'une configuration initiale spécifique, représentant un niveau de charge donné dans le système. Pour chaque scénario, nous mesurons la durée (nombre d'étapes) et l'amplitude (nombre de cellules impliquées) de l'avalanche résultante, en comparant les comportements avec et sans le mécanisme de seuil critique dynamique.

Les scénarios sont d'abord présentés, puis les résultats analysés.

5.2.2.1 Présentation des scénarios

Comme le mécanisme de seuil critique dynamique ne concerne pas uniquement le tamis mais s'applique au modèle du tas de sable en général, trois des quatre scénarios proposés se déroulent dans un tas de sable canonique de taille $128 \times 128 = 16\,384$ cellules.

Scénario 1 Le système est au bord de l'éboulement généralisé : chaque cellule est initialisée avec exactement 3 grains. Un grain supplémentaire est ensuite ajouté à une cellule choisie aléatoirement afin de déclencher une avalanche.

Scénario 2 Le système est initialisé dans un état critique : chaque cellule reçoit aléatoirement entre 3 et 5 grains, suivant une distribution uniforme (moyenne de 4 grains). Environ $\frac{2}{3}$ des cellules se trouvent donc en état critique. Aucun grain supplémentaire n'est ajouté, l'avalanche pouvant être déclenchée spontanément du fait de l'instabilité globale.

Scénario 3 Le système est dans un état plus modéré : chaque cellule est initialisée avec 2 ou 3 grains de façon aléatoire et uniforme. Six grains supplémentaires sont ensuite ajoutés aléatoirement pour déclencher une avalanche. Ce scénario illustre un cas plus proche d'un fonctionnement opérationnel courant.

Scénario 4 Le système est fermé et largement saturé : la structure de la grille est rendue torique, supprimant ainsi les bordures et uniformisant le nombre de voisins pour chaque cellule. Dans ce cas, il devient impossible d'évacuer les grains en excès vers l'extérieur, ce qui empêche une dissipation naturelle de l'instabilité. Cette configuration simule le comportement du tamis dans un espace limité, ce qui nous intéresse particulièrement puisque, dans ce contexte, le tamisage n'intervient qu'après la fin des avalanches.

Chaque cellule est initialisée avec entre 4 et 10 grains (distribution uniforme). Dans une telle configuration, le modèle canonique (sans seuil dynamique) ne peut atteindre la stabilité, toutes les cellules étant en état critique. Ce scénario vise donc à évaluer la capacité du seuil dynamique à stabiliser une configuration qui, autrement, conduirait à des avalanches infinies.

5.2.2.2 Résultats des scénarios

La Figure 5.2 propose une comparaison de la dynamique des avalanches d'un tas de sable canonique avec et sans seuil critique dynamique pour les scénario 1 à 3. Dans l'ensemble de ces configurations, l'introduction du seuil dynamique permet de réduire significativement l'ampleur (nombre de cellules impliquées) ainsi que la durée des avalanches.

Ce phénomène est confirmé par la Figure 5.3, qui présente la distribution des durées d'avalanches sur une simulation de 400 000 cycles. Bien que la distribution globale soit peu altérée, ce qui traduit une auto-organisation toujours fonctionnelle, on observe une légère augmentation des avalanches de durée moyenne, ce qui traduit un meilleur amortissement des pics critiques.

Le Tableau 5.1 propose une analyse quantitative complémentaire, en comparant non seulement les durées des avalanches, mais aussi les seuils critiques moyens atteints à la fin de chacune d'entre elles, pour les scénarios évoqués. Le quatrième scénario y est inclus. Dans la configuration du quatrième scénario, l'introduction du seuil critique dynamique permet au système de se stabiliser en 1450 itérations, alors que le modèle canonique ne peut pas atteindre une stabilité. De manière notable, dans toutes les situations simulées, le seuil critique moyen final observé est supérieur de quasiment deux unités à la charge moyenne initiale des cellules.

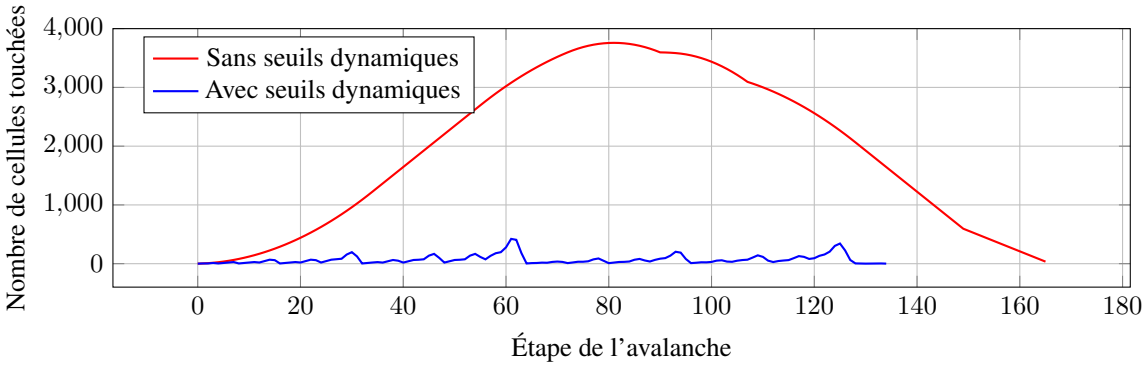
Seuil critique dynamique	**Scénario 1**		**Scénario 2**		**Scénario 3**		**Scénario 4**	
	Durée	*Seuil*	*Durée*	*Seuil*	*Durée*	*Seuil*	*Durée*	*Seuil*
Sans	165	4	7145	4	467	4	Infinie	4
Avec	134	4,9	161	5,8	232	4,25	1450	8,8

TABLE 5.1 – Durée de l'avalanche et seuil critique moyen des cellules d'un tas de sable canonique de taille 128 pour les trois scénarios de la Figure 5.2. Un quatrième scénario est proposé : la structure est rendue toroïdale (disparition des bords pour éjecter les grains) et les cellules sont remplie de 4 à 10 grains chacune.

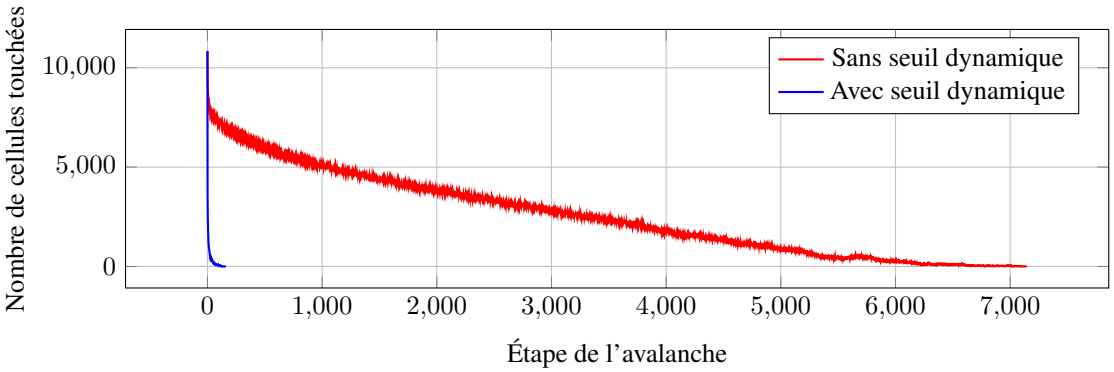
En définitive, le mécanisme simple qu'est-ce le seuil critique dynamique constitue une solution efficace pour contenir les avalanches de grande ampleur tout en préservant la dynamique d'auto-organisation du système. Cependant, cette stratégie d'élévation des seuils critiques ne fait que retarder le problème fondamental de la surcharge dans le tamis : sans évacuation des grains adéquat, les seuils ne feront qu'augmenter indéfiniment, menaçant à terme l'efficacité du système. Nous aborderons dans les sections suivantes les approches d'adaptation complémentaires concernant les capacités de tamisage des cellules, visant à évacuer dynamiquement les excès de charge de manière autonome.

5.3 Modélisation de la capacité de tamisage dynamique

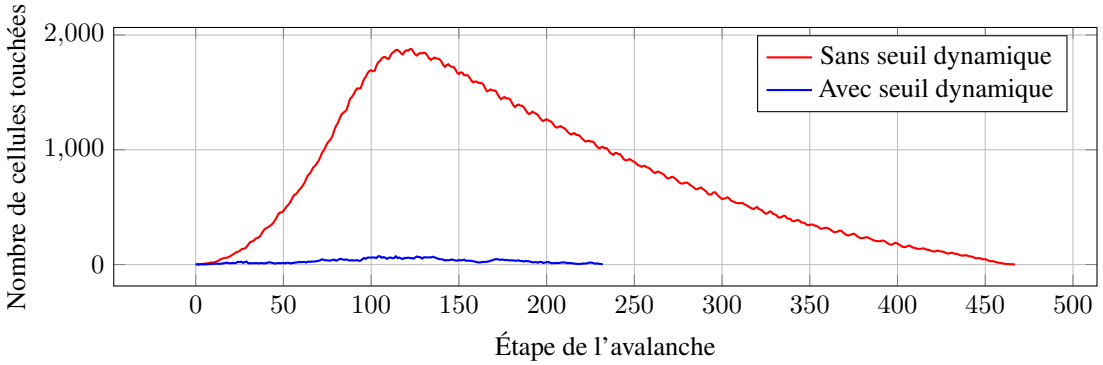
Jusqu'à présent, la capacité de tamisage des cellules était définie de manière fixe et invariable. Dans le modèle présenté en Section 3.5.3, les cellules dépourvues de grains sont considérées comme éteintes, leur capacité de tamisage étant alors nulle. Ce mécanisme est conservé dans les modèles adaptatifs étudiés dans les sections suivantes. Toutefois, afin de permettre l'évolution dynamique des capacités, il devient nécessaire de distinguer deux notions fondamentales : la capacité latente de tamisage, sujette à adaptation, et la capacité



(a) Scénario 1 : toutes les cellules sont proches de l'éboulement (3 grains) et un grain est ajouté à une cellule aléatoire. Le nombre de cellules touchées par l'avalanche est drastiquement réduit, en plus d'avoir une durée plus faible, grâce au seuil dynamique.



(b) Scénario 2 : les cellules ont de 3 à 5 grains (de manière aléatoire et uniforme) ; $\frac{2}{3}$ des cellules sont en état critique. L'avalanche de très grande ampleur est très rapidement amortie (161 itérations) et le nombre de cellules touchées par l'avalanche est drastiquement réduit grâce au seuil dynamique.



(c) Scénario 3 : les cellules sont initialisées avec 2 ou 3 grains de manière aléatoire et uniforme, et six grains sont ajoutés à des cellules aléatoires pour déclencher une avalanche. De même que pour le scénario 1, le nombre de cellules impactées est très largement réduit et la durée de l'avalanche est ici divisée par deux grâce au seuil critique dynamique.

FIGURE 5.2 – Effet du seuil dynamique sur la dynamique des avalanches dans un tas de sable canonique de taille 128. Trois scénarios sont proposés : (a) toutes les cellules sont au bord de l'éboulement et 1 grain est déposé pour créer de l'instabilité ; (b) $\frac{2}{3}$ des cellules sont dans un état critique ; (c) les cellules sont initialisées avec 2 ou 3 grains de manière aléatoire et uniforme, puis six grains sont ajoutés pour provoquer une avalanche.

effective, représentant la "puissance réelle" de traitement exercée à un instant donné.

Chaque cellule est ainsi caractérisée par une capacité latente de tamisage, qui évolue de manière dynamique au fil du temps selon des règles d'adaptation locales, indépendamment de l'état d'occupation de la cellule. Cette

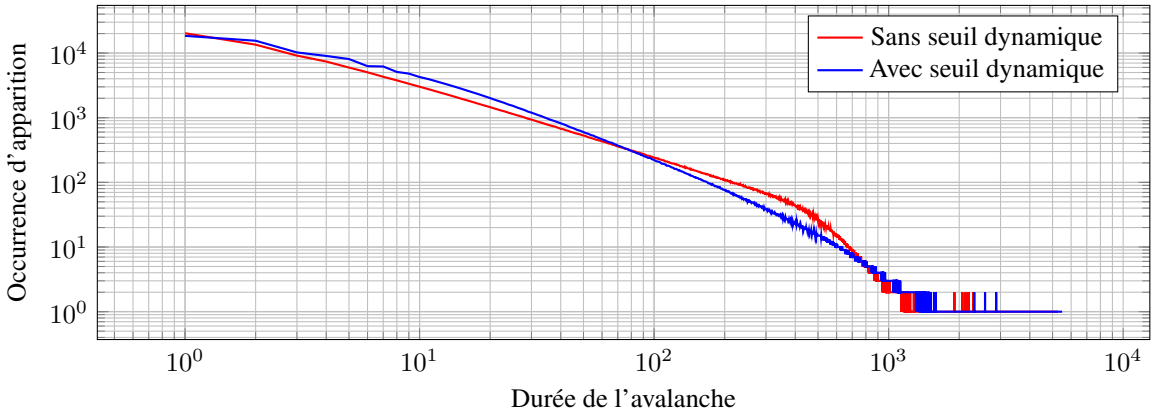


FIGURE 5.3 – Distribution des durée des avalanches dans un tas de sable canonique de taille 128 pour une simulation de 400 000 cycles. La distribution n'est que légèrement modifiée avec le seuil dynamique.

capacité latente représente le potentiel intrinsèque de traitement de la cellule. Lorsqu'une cellule reçoit un grain, sa capacité latente devient active, donnant lieu à une capacité effective de tamisage correspondant à l'intensité réelle du traitement exercé.

En l'absence de grain, la capacité effective de tamisage est nulle : bien que la capacité latente soit présente, elle n'est pas mobilisée. La valeur minimale de la capacité latente est fixée à 1, traduisant l'activation automatique d'une cellule dès réception d'un grain, qu'il y ait eu adaptation préalable ou non. Cette modélisation permet de reproduire fidèlement le comportement du tamis présenté précédemment : dans ce cas, la capacité latente est constante (valeur 1), et la capacité effective est égale à 1 si la cellule est occupée, sinon 0.

Cette distinction entre capacité latente et capacité effective présente deux avantages majeurs :

1. **Indépendance vis-à-vis de l'occupation :** la capacité latente peut évoluer même en l'absence de grains, ce qui permet aux cellules de conserver et faire croître leur potentiel de traitement pour les phases futures d'occupation, sans réinitialisation systématique au minimum lors de la libération d'un grain.
2. **Estimation énergétique locale :** la capacité effective constitue un indicateur de la consommation énergétique réelle associée au traitement des grains. À l'inverse, la capacité latente représente la consommation potentielle si la cellule était pleinement sollicitée.

Cette distinction sera exploitée dans la suite du chapitre, où la capacité effective de tamisage sera utilisée comme mesure de performance énergétique locale des différents modèles d'adaptation proposés.

5.4 Adaptation des capacités par entropie locale

Le principe de l'adaptation par entropie locale est de monitorer le mouvement des grains à l'échelle des cellules du tamis, afin que chacune prenne une décision quant à sa capacité de tamisage. Le but est de détecter une surcharge (impliquant un mouvement conséquent) pour que les cellules adaptent leur capacité de tamisage de manière décentralisée. Une étape de mise à jour des capacités vient alors se positionner à chaque cycle entre la gestion de l'avalanche et le tamisage des grains, comme proposé en pseudo-code dans l'Algorithme 6.

Nous allons d'abord discuter de l'objectif de cette entropie et de la manière dont elle se mesure, puis nous explorerons deux utilisations qui en sont faites pour adapter les capacités de tamisage des cellules. Nous verrons ensuite comment sont déterminés les paramètres de ces approches. Enfin, nous les comparerons pour n'utiliser que la meilleure pour la suite du chapitre.

Algorithm 6: Processus de simulation du tamis auto-adaptatif par entropie locale

Input: G : grille
cycles : nombre de cycles de simulation

```
1  cycle ← 0
2  while cycle < cycles do
3  |   Étape 1 : dépôt d'un grain sur une cellule aléatoire de G
4  |   Étape 2 : gestion de l'avalanche potentielle
5  |
   |   /* Auto-adaptation des cellules                                  */
6  |   foreach cellule ∈ G do
7  |   |   Calcul de l'entropie locale de cellule
8  |   |   Mise à jour de la capacité latente de tamisage de cellule
9  |   end
10 |
11 |   Étape 4 : tamisage des grains
12 |   cycle ← cycle + 1
13 end
```

5.4.1 L'entropie locale

Ce que nous qualifions d'entropie dans le modèle du tamis correspond à une mesure du mouvement des grains au sein du système. Une entropie faible traduit un système organisé, dans lequel les grains parviennent naturellement à s'équilibrer sans nécessiter d'avalanches. À l'inverse, une entropie élevée signale un désordre transitoire, où les grains doivent être redistribués par des avalanches pour atteindre un état stable. L'entropie constitue donc une métrique pertinente pour détecter les situations de surcharge, caractérisées par une intensification du mouvement granulaire.

Afin d'établir une valeur de référence, nous nous appuyons sur le modèle du tas de sable canonique, reconnu pour sa capacité à maintenir un équilibre auto-organisé à la limite du chaos. Ce système évacue les grains dès que nécessaire, empêchant toute surcharge prolongée, tout en conservant une dynamique granulaire suffisante pour l'auto-organisation. Nous définissons ainsi une entropie de référence *pré-surcharge* à partir des cellules les plus sollicitées (typiquement celles au centre du tas de sable). Cette entropie est exprimée comme le ratio entre le nombre total de grains reçus et la durée de la simulation :

$$E_{ref} = \frac{\text{nombre de grains tombés}}{\text{durée de la simulation}}$$

Elle représente la fréquence moyenne de chute des grains sur une cellule par cycle. Dans le cadre du modèle canonique, cette valeur atteint : $E_{ref} \simeq 0{,}3$.

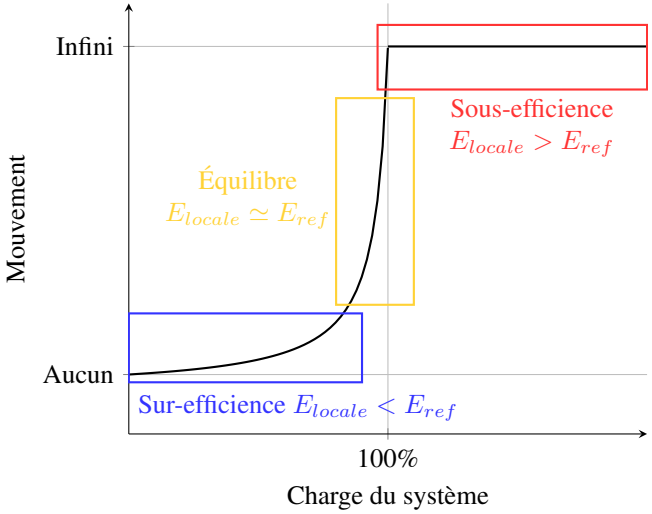


FIGURE 5.4 – Illustration des différents états du système en fonction du mouvement par rapport à la comparaison des entropies locale et de référence.

Dans le tamis auto-adaptatif, chaque cellule effectue un suivi local de son entropie au fil du temps. Elle enregistre, sur une fenêtre glissante des n derniers cycles, le nombre de grains qu'elle a reçus, et calcule sa valeur d'entropie locale selon :

$$E_{locale} = \frac{\text{nombre de grains enregistrés}}{n}$$

Une valeur de $E_{locale} = 1$ indique qu'en moyenne, la cellule reçoit un grain à chaque cycle. Nous désignerons cette fenêtre d'observation comme la **fenêtre d'entropie**.

La comparaison entre E_{locale} et E_{ref} permet alors d'ajuster dynamiquement les capacités de tamisage :

— Si $E_{locale} > E_{ref}$, la cellule est soumise à une activité anormalement élevée, symptôme d'une surcharge locale. Il est donc nécessaire d'augmenter sa capacité de tamisage pour faciliter l'évacuation des grains.

— Inversement, si $E_{locale} < E_{ref}$, la cellule participe insuffisamment à l'auto-organisation, ce qui peut signaler une surcapacité. Dans ce cas, réduire sa capacité permet de réengager cette cellule dans la dynamique d'équilibrage du système.

L'entropie de référence joue donc le rôle de cible idéale : elle incarne le compromis entre l'auto-organisation nécessaire à l'équilibrage du système et le tamisage global, assurant un fonctionnement optimal du tamis auto-adaptatif. La Figure 5.4 illustre visuellement ces différents scénarios, mettant en évidence les zones de sous- et de sur-utilisation par rapport à cette référence.

5.4.2 Méthode naïve

Nous introduisons ici une version naïve d'adaptation utilisant l'entropie locale définie précédemment. L'objectif de cette approche est d'offrir une première implémentation simple et accessible, permettant d'évaluer le potentiel de la stratégie d'adaptation sans recourir à des mécanismes complexes.

Cette version, baptisée méthode “+1 -1”, repose sur une règle élémentaire : chaque cellule décide, à chaque cycle, d’augmenter ou de diminuer sa capacité latente de tamisage en fonction de sa propre entropie locale. Trois cas de figure sont distingués :

— $E_{locale} > E_{ref}$: la cellule incrémente sa capacité latente de 1, en réponse à une activité granulaire supérieure à la normale ;

— $E_{locale} < E_{ref}$: la cellule décrémente sa capacité latente de 1 (avec un minimum fixé à 1), reflétant une sous-utilisation locale ;

— $E_{locale} = E_{ref}$: la cellule est en équilibre, aucune modification n’est apportée.

Le pseudo-code de ce mécanisme est présenté dans l’Algorithme 7.

Algorithm 7: Adaptation par entropie locale : méthode “+1 -1”

Input: *cellule* : cellule à adapter
E_{ref} : entropie de référence

```
1 E_locale ← cellule.entropie()
2 if E_locale > E_ref then
3 |   cellule.capacitéLatente ← cellule.capacitéLatente + 1
4 else if E_locale < E_ref then
5 |   cellule.capacitéLatente ← max(1, cellule.capacitéLatente - 1)
6 end
```

L’application de cette stratégie engendre une variation en “vague” de la capacité latente d’une cellule au fil du temps, comme l’illustre la Figure 5.5. Par exemple, pour une fenêtre d’entropie de taille 10, la capacité n’augmente que lorsque plus de 3 grains sont observés dans cette fenêtre ($E_{locale} > E_{ref}$), et décroît lorsque ce nombre tombe en dessous de 3 ($E_{locale} < E_{ref}$).

Cette dynamique locale, appliquée à l’échelle globale du tamis, permet une adaptation collective : plusieurs cellules ajustent simultanément leurs capacités pour évacuer les grains responsables d’une surcharge ponctuelle. Cette réaction coordonnée favorise un retour à l’équilibre dans les zones instables. Nous verrons, dans les résultats présentés en fin de section, que ce mécanisme permet au tamis de s’ajuster efficacement et que la capacité de tamisage moyenne des cellules suit de près la quantité de grains injectée à chaque instant.

5.4.3 Méthode proportionnelle

La seconde stratégie d’adaptation reposant sur l’entropie locale en affine l’exploitation en y intégrant une notion de proportionnalité. Contrairement à la méthode “+1 -1” qui ajuste la capacité latente par paliers discrets, cette version ajuste directement la capacité à la quantité de grains effectivement observée dans la fenêtre d’entropie.

Plus précisément, lorsque l’entropie locale dépasse la référence, la cellule adopte comme nouvelle capacité latente le nombre de grains enregistrés sur la fenêtre d’observation. Ce mécanisme permet une adaptation plus rapide et plus précise, en alignant immédiatement la capacité au niveau de stress local observé, tout en stabilisant cette valeur aussi longtemps que l’activité reste cohérente. Ainsi, la capacité peut légèrement décroître si le nombre de grains diminue, tout en restant supérieur au seuil de déclenchement.

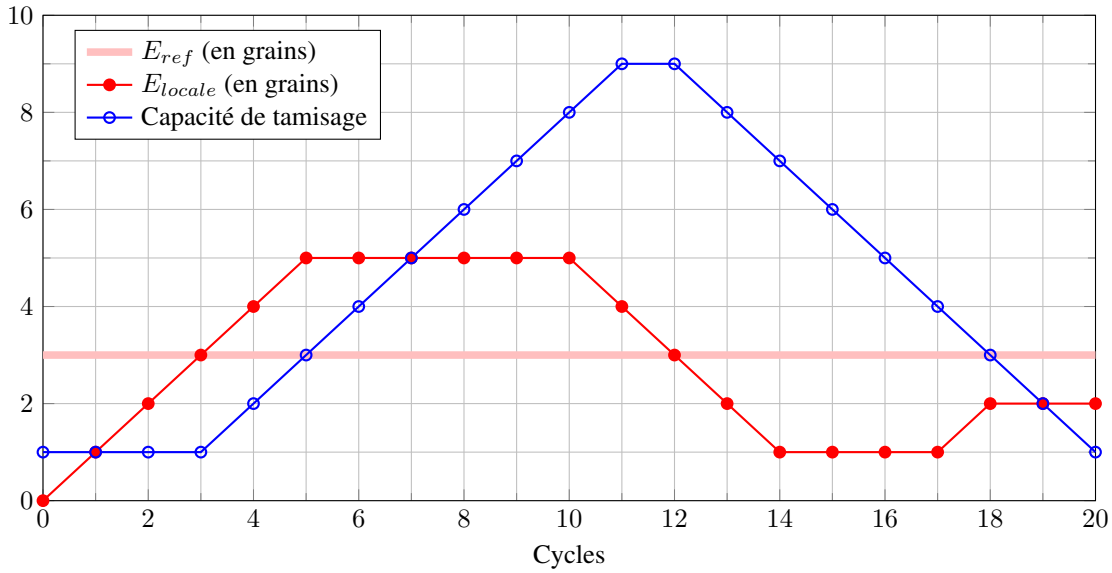


FIGURE 5.5 – Exemple de l'adaptation naïve par entropie locale d'une cellule avec une fenêtre d'entropie de taille 10. Lorsque l'entropie locale (courbe rouge) dépasse la référence de 3 grains (barre rose ; $E_{ref} = \frac{3}{10} = 0{,}3$), la capacité de la cellule (courbe bleue) augmente de 1 à chaque cycle jusqu'à ce que l'entropie locale retourne en dessous de la référence. La capacité diminue alors, produisant une réaction en "vague".

La réduction des capacités, quant à elle, obéit à une logique distincte. Plutôt que d'opérer des décréments progressifs, la cellule réinitialise sa capacité latente dès qu'elle devient vide, c'est-à-dire lorsqu'aucun grain ne transite par elle. Cette approche offre un délai naturel de vidange, permettant aux cellules de résorber la surcharge avant de retrouver un mode de fonctionnement minimal. On peut faire ici le parallèle avec les systèmes de traitement de tâches : une ressource (cellule) qui n'a plus de file d'attente (grains) revient à un état énergétiquement sobre, avec une capacité minimale (fixée à 1 dans notre étude).

La stratégie peut donc se résumer par les trois règles suivantes :

— $E_{locale} > E_{ref}$: la cellule fixe sa capacité latente au nombre de grains enregistrés dans sa fenêtre d'entropie ;

— $E_{locale} \leq E_{ref}$: la cellule conserve sa capacité latente actuelle ;

— Cellule vide : la cellule réinitialise sa capacité latente au minimum fixé à 1.

Le pseudo-code correspondant est présenté dans l'Algorithme 8.

Algorithm 8: Adaptation par entropie locale : méthode proportionnelle

Input: *cellule* : cellule à adapter
E_{ref} : entropie de référence

```
E_locale ← cellule.entropie()
if cellule.grains = 0 then
    cellule.capacitéLatente ← 1
else if E_locale > E_ref then
    cellule.capacitéLatente ← Σ_{i=0}^{n} cellule.fenêtreEntropie[i]
end
```

À la différence de la méthode "+1 -1", cette version proportionnelle induit une évolution contrôlée et ré-

active des capacités latentes, comme illustré en Figure 5.6. La capacité augmente instantanément à un niveau modéré lorsque $E_{locale} > E_{ref}$, puis se maintient stable tant que la surcharge persiste. Dès que la cellule se vide, elle retrouve immédiatement sa capacité minimale, prête à répondre à une nouvelle sollicitation.

Ce cycle d'augmentation et de réinitialisation, appliqué collectivement, permet au tamis de maintenir une capacité effective moyenne étroitement corrélée à la proportion de grains injectée. Nous montrerons dans la suite de cette section que cette stratégie assure une adaptation fluide et efficace du système.

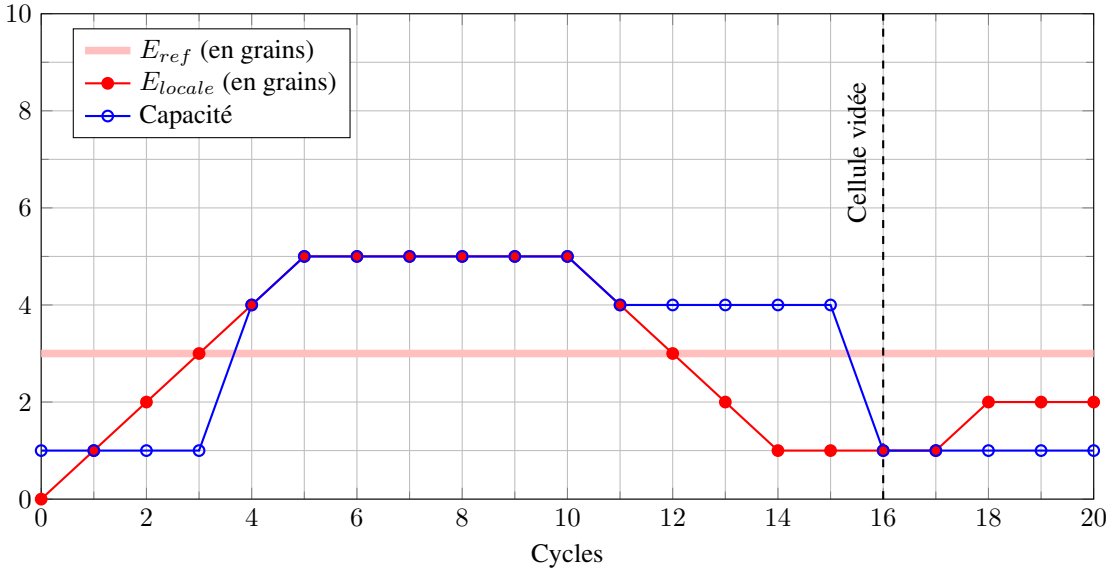


FIGURE 5.6 – Exemple de l'adaptation proportionnelle par entropie locale d'une cellule avec une fenêtre d'entropie de taille 10. Lorsque l'entropie locale (courbe rouge) dépasse la référence de 3 grains (barre rose ; $E_{ref} = \frac{3}{10} = 0{,}3$), la capacité de la cellule (courbe bleue) se fixe au nombre de grains enregistrés à chaque cycle jusqu'à ce que l'entropie locale retourne en dessous de la référence. La capacité est réinitialisée lorsque la cellule devient vide au cycle 16.

5.4.4 Détermination des paramètres de l'entropie locale

Les stratégies d'adaptation fondées sur l'entropie locale reposent sur deux paramètres essentiels : l'entropie de référence et la taille de la fenêtre d'observation (ou fenêtre d'entropie). Le premier, initialement fixé à E_{ref}, découle d'une estimation empirique réalisée à partir du modèle canonique du tas de sable. Toutefois, cette valeur de référence n'est pas universellement optimale et peut varier selon la taille de la fenêtre utilisée pour le calcul de l'entropie locale.

Le choix de ces deux paramètres conditionne l'équilibre global du système entre trois objectifs cruciaux :

— une capacité de tamisage appropriée face aux sollicitations ;

— une consommation énergétique contenue, en évitant des surcapacités inutiles ;

— une répartition homogène de l'activité entre les cellules, évitant que certaines deviennent des points de congestion ou de faiblesse.

En effet, dans un système décentralisé tel que le tamis, il est souhaitable que l'ensemble des cellules soient sollicitées de manière équitable. Une cellule surdimensionnée par rapport aux autres agirait comme un puits,

absorbant une part disproportionnée des grains, ce qui l'expose davantage à l'usure ou à des pannes, et nuit à la robustesse globale du système.

Afin de calibrer les stratégies d'auto-adaptation par entropie locale, il est donc nécessaire d'explorer un espace de valeurs pour ces deux paramètres, en évaluant leur influence sur le comportement du système. Cette étude paramétrique vise non seulement à optimiser les performances de chaque stratégie, mais aussi à garantir des conditions équitables de comparaison entre les différentes approches d'adaptation proposées dans ce travail.

5.4.4.1 Cadre d'étude

Afin d'évaluer l'impact des paramètres d'adaptation, nous menons une étude systémique des performances du tamis auto-adaptatif en faisant varier les deux composantes clés : l'entropie de référence E_{ref} et la taille de la fenêtre d'entropie. L'entropie de référence est explorée autour de sa valeur canonique ($0{,}3$), dans un intervalle allant de $0{,}05$ à $0{,}45$ par pas de $0{,}05$. Quant à la fenêtre d'entropie, sa taille varie de 5 à 20 cycles observés. Des fenêtres plus longues introduiraient une inertie excessive dans la détection des changements, ce qui limiterait la réactivité du système aux déséquilibres locaux.

L'objectif de cette étude est d'identifier les couples $\{E_{ref}$; fenêtre d'entropie$\}$ qui permettent une réaction juste et maîtrisée du système face aux sollicitations, en assurant un compromis entre réactivité, efficacité énergétique et égalité d'utilisation des cellules. Chaque couple de paramètres est évalué selon quatre critères principaux :

- **Capacité de tamisage moyenne :** moyenne, sur toute la simulation, des capacités effectives de tamisage des cellules. Elle doit se rapprocher de la proportion de charge injectée dans le système afin d'assurer un équilibre sans surdimensionnement inutile.
- **Capacité de tamisage maximale moyenne :** moyenne des capacités effectives maximales de tamisage observée parmi les cellules pendant la simulation. Une valeur élevée indique une disparité de la réaction des cellules, ce qui va à l'encontre de l'uniformisation recherchée.
- **Nombre d'avalanches "infinies" :** une avalanche est considérée comme infinie si elle excède 500 cycles, seuil basé sur la durée maximale observée dans un tas de sable de même dimension que le tamis, augmenté d'une marge de sécurité. Une telle avalanche est interrompue pour préserver la progression de la simulation.
- **Consommation énergétique :** somme des mouvements de grains provoqués par les avalanches et des capacités de tamisage effectives des cellules au cours de la simulation, exprimée en pourcentage de la consommation optimale. Cette consommation optimale est celle d'un système parfaitement statique, sans mouvement, et distribuant équitablement la charge.

L'évaluation repose sur une série de simulations durant chacune 100 000 cycles, précédés d'une phase d'initialisation de 10 000 cycles. Chaque scénario est répété 100 fois afin de lisser les variations dues à l'aléa. Le tamis étudié est de taille fixe ($32 \times 32 = 1024$ cellules), et à chaque cycle, un grain est injecté aléatoirement dans la grille. Deux types de politiques d'injection de charge sont testés :

- **Politiques fixes :** la taille des grains reste constante pendant la simulation, choisie parmi 2048, 3072 et

4096, soit 2 à 4 fois la taille du tamis.

— **Politiques fluctuantes** :

- **Charge sinusoïdale :** variation périodique entre 2048 et 4096, sur une période de 50 000 cycles.
- **Charge aléatoire avec pics :** base gaussienne centrée sur 2048 avec un écart-type de 768, associée à des pics brusques.

Pour cette dernière politique, la taille minimale des grains est bornée à 1 pour éviter des charges nulles ou négatives. Chaque cycle a une probabilité $P_{pic} = 2{,}5 \cdot 10^{-3}$ de déclencher un pic de charge durant 50 à 150 cycles. Ces pics consistent en une charge uniforme aléatoire entre 9216 et 11264, soit de 9 à 11 fois la taille du système. La Figure 5.7 illustre une réalisation typique sur 10 000 cycles. La taille moyenne des grains sous cette politique est de 3633,4, ce qui implique une capacité optimale par cellule de $\frac{3633{,}4}{1024} \simeq 3{,}55$. Pour garantir l'équité de la comparaison entre stratégies, une même instance de charge aléatoire est utilisée pour chaque méthode évaluée.

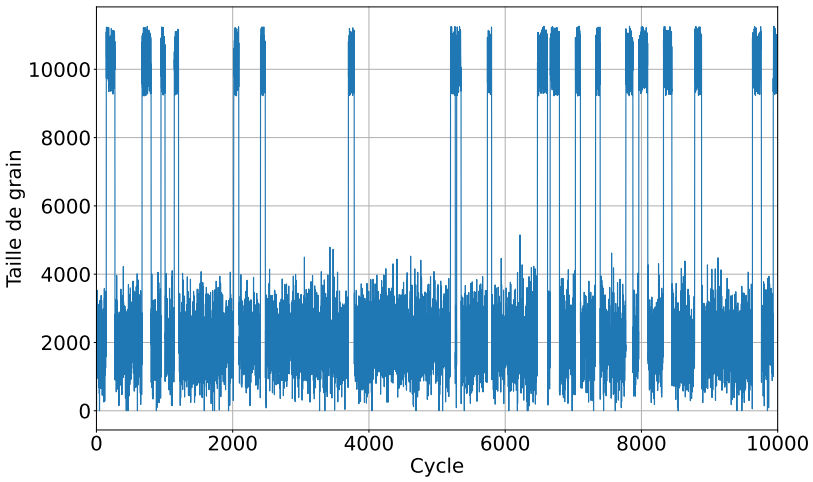


FIGURE 5.7 – Exemple des tailles de grain générées avec la politique aléatoire sur 10000 cycles.

Enfin, la capacité latente minimale de chaque cellule est fixée à 1, et toute cellule inactive (ne traitant aucun grain) entre en mode veille, sans consommation d'énergie (capacité effective). La consommation énergétique optimale pour une simulation donnée est définie comme :

$$consommation = \text{capacité cellule optimale} \times \text{nombre de cellules} \times \text{durée de simulation}$$

Par exemple :

— Pour une taille de grain constante de 2048 : $consommation = 2{,}048 \cdot 10^8$.

— Pour la politique aléatoire avec pics : $consommation = 3{,}633 \cdot 10^8$.

Comme nous le verrons ci-après, les modèles d'adaptation ajustent les capacités de tamisage au nécessaire. La consommation induite par le tamisage correspond donc à la consommation optimale. Les différences dans la consommation totale ne correspondent alors qu'à la consommation associée aux avalanches (déplacements

des grains). Nous conservons tout de même l'implication du tamisage afin de pouvoir, dans le futur, comparer ces modèles avec d'autres qui proposeront des résultats d'adaptation potentiellement différents.

5.4.4.2 Adaptation naïve

Les premiers résultats indiquent que l'adaptation naïve fonctionne de manière systématique, indépendamment du couple de paramètres $\{E_{ref}$; fenêtre d'entropie$\}$ utilisé. La capacité moyenne de tamisage des cellules s'ajuste correctement à la taille des grains injectés : elle atteint approximativement 2 pour des grains de taille 2048, 3 pour 3072, et 4 pour 4096. Le Tableau 5.2 résume ces résultats en indiquant, pour chaque taille de grain, les valeurs minimales et maximales de capacité moyenne atteinte pour l'ensemble des couples. L'écart observé est très faible, ce qui traduit une robustesse du modèle d'adaptation naïve par entropie locale, quel que soit le réglage de ses paramètres.

Taille de grain	**Capacité moyenne minimale**		**Capacité moyenne maximale**	
	Seuil fixe	*Seuil dynamique*	*Seuil fixe*	*Seuil dynamique*
2048	1,999	1,977	2	2
3072	2,999	2,965	3,001	3
4096	3,999	3,953	4,002	4

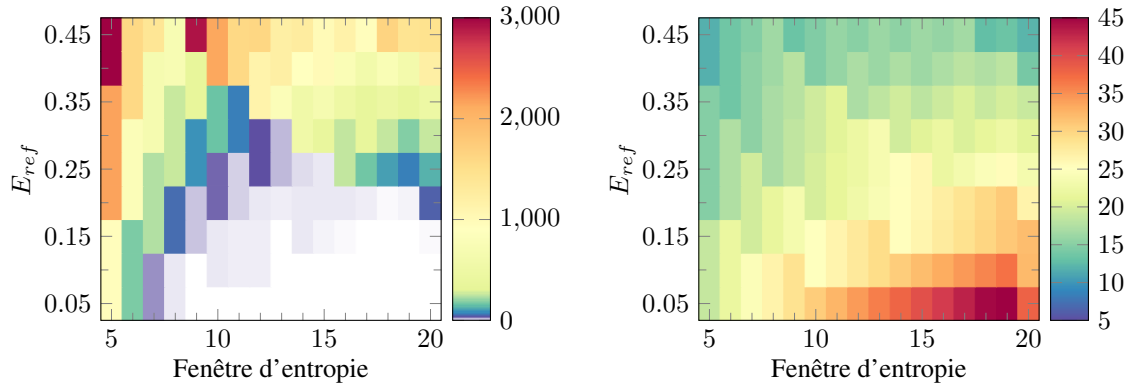
TABLE 5.2 – Capacité moyenne atteinte par les couples $\{E_{ref}$; fenêtre d'entropie$\}$ selon la taille des grains injectés dans le tamis avec adaptation naïve par entropie locale, avec et sans seuil critique dynamique. Pour chaque taille, les valeurs minimales et maximales des couples sont proposées.

Cependant, au-delà de la faisabilité de l'adaptation, la qualité des performances varie considérablement d'un couple de paramètres à l'autre. Nous nous concentrons donc sur le cas le plus contraignant, correspondant à une taille de grain de 4096, afin d'évaluer finement la pertinence des réglages. La Figure 5.8 propose deux cartes de chaleur pour chaque couple de paramètres : la capacité maximale atteinte par des cellules et le nombre d'avalanches infinies survenues. Une tendance se dégage des résultats : tous les couples pour lesquels $E_{ref} \geq 0{,}25$ présentent des avalanches infinies, tandis que les capacités maximales atteintes sont les plus élevées pour les couples ayant $E_{ref} < 0{,}25$.

Cette opposition illustre une forme d'inertie excessive dans le mécanisme d'adaptation lorsque l'entropie de référence est faible, combinée à une fenêtre d'observation large. Dans ce cas, très peu de grains suffisent à déclencher une augmentation de capacité, et ces événements étant étalés sur une longue fenêtre, ils s'accumulent durablement, même si la situation se stabilise.

Par ailleurs, bien que la capacité moyenne du système se stabilise autour de 4, les pics locaux peuvent être extrêmement hétérogènes. Certaines cellules atteignent des valeurs jusqu'à 11 fois supérieures à l'objectif, tandis qu'une majorité reste au minimum. Ainsi, les couples viables (sans avalanche infinie) ne le sont qu'au prix d'une surcapacité ponctuelle extrême, ce qui révèle un déséquilibre dans la répartition de l'effort de tamisage.

L'introduction du seuil critique dynamique (en plus de l'adaptation des capacités de tamisage) ne remet pas en cause la capacité du système à s'adapter : la capacité moyenne reste alignée avec la taille des grains injectés, comme le confirme de nouveau le Tableau 5.2.



(a) Nombre d'avalanches infinies survenues.

(b) Capacité maximale atteinte.

FIGURE 5.8 – Résultats de l'adaptation naïve par entropie locale pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains à quatre fois la taille du système.

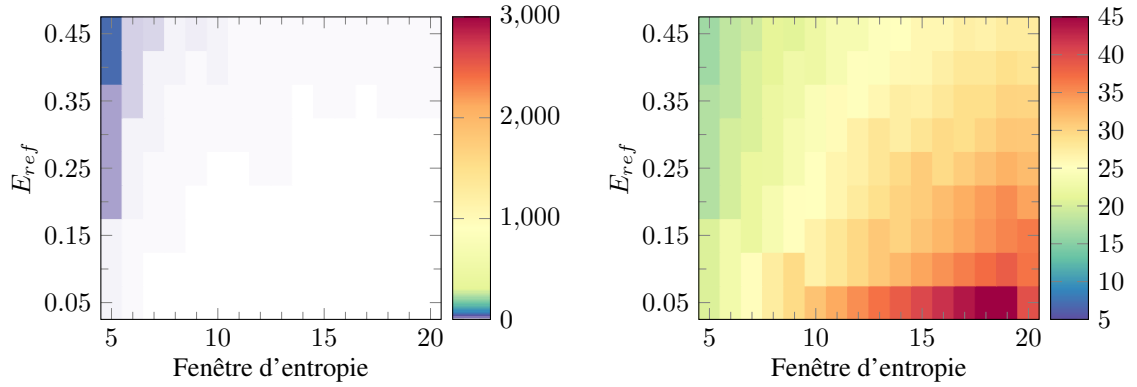
La Figure 5.9 compare alors les performances des couples de paramètres avec cette nouvelle stratégie, selon les deux mêmes critères que précédemment. Trois observations ressortent :

— le nombre d'avalanches infinies a drastiquement diminué de manière générale ;

— davantage de couples deviennent viables, c'est-à-dire exempts d'avalanches infinies ; les autres en présentent très peu ;

— la capacité maximale atteinte augmente globalement, en particulier lorsque la fenêtre d'entropie est large.

Ces résultats traduisent un gain de robustesse important grâce à l'adaptation conjointe des capacités et des seuils, malgré une légère hausse des valeurs maximales de capacité. Comme dans le cas sans seuil dynamique, les couples viables correspondent à ceux ayant les capacités maximales les plus élevées, indiquant que la résilience du système repose toujours en partie sur une surcapacité locale.



(a) Nombre d'avalanches infinies survenues.

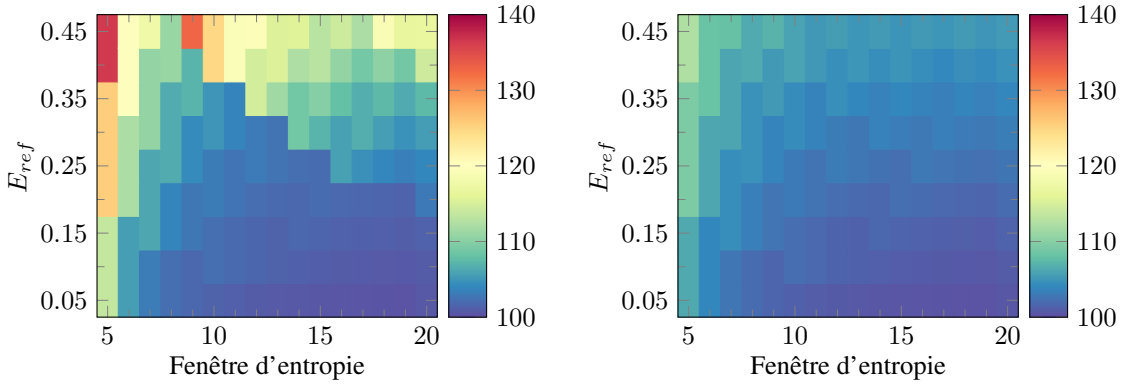
(b) Capacité maximale atteinte.

FIGURE 5.9 – Résultats de l'adaptation naïve par entropie locale avec seuil critique dynamique pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains à quatre fois la taille du système.

Notons que les couples avec une fenêtre d'entropie de taille 5 restent systématiquement non viables, malgré l'ajout du seuil critique dynamique. Ce comportement s'explique par une durée de réaction faible du système.

La courte fenêtre empêche l'accumulation d'observations significatives, rendant l'augmentation des capacités plus difficile. En revanche, la diminution de capacité reste très sensible aux variations ponctuelles. Ce déséquilibre se reflète dans les mesures de capacité maximale : les cellules fluctuent trop rapidement, sans atteindre des valeurs suffisantes pour compenser les pics de charge, ce qui ralentit l'évacuation des grains et accroît l'instabilité.

Enfin, la Figure 5.10 présente la consommation énergétique du système avec et sans seuil critique dynamique. L'ajout de l'adaptation des seuils permet de réduire considérablement le coût énergétique associé aux mouvements de grains, rapprochant la majorité des couples viables de la consommation optimale théorique. Les couples déjà viables sans seuil dynamique ne sont que peu affectés, ce qui souligne leur efficacité intrinsèque dans l'équilibrage entre tamisage et auto-organisation.



(a) Sans seuil dynamique.

(b) Avec seuil dynamique.

FIGURE 5.10 – Consommation énergétique du tamis utilisant l'adaptation naïve par entropie locale, avec et sans seuil dynamique, pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains à quatre fois la taille du système. La consommation est exprimée en pourcentages par rapport à la consommation optimale.

Sur la base de l'ensemble des critères d'évaluation (absence d'avalanches infinies, faible capacité maximale, faible consommation), plusieurs couples de paramètres se démarquent. Le couple $\{0{,}25\,;\,11\}$ a été retenu comme référence pour la suite des comparaisons. Il est robuste, présente l'une des capacités maximale les plus faibles parmi les couples viables et présente une consommation maîtrisée. Ce couple sera utilisé comme point de comparaison entre les deux méthodes d'auto-adaptation par entropie locale développées dans cette étude.

5.4.4.3 Adaptation proportionnelle

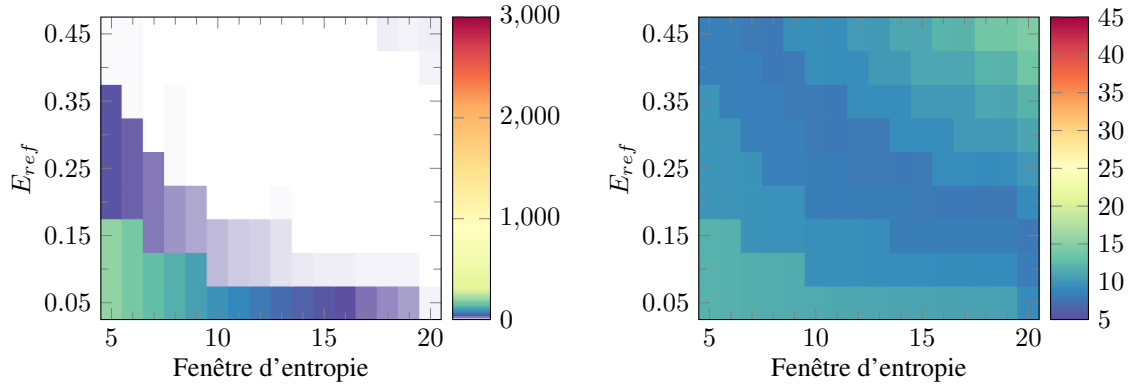
Comme pour la méthode naïve, l'auto-adaptation proportionnelle se révèle pleinement fonctionnelle, comme en témoignent les résultats du Tableau 5.3. La capacité moyenne de tamisage des cellules augmente avec la taille des grains injectés dans le système, et les variations entre les couples de paramètres sont faibles. Cela indique une robustesse du mécanisme d'adaptation proportionnelle, quelles que soient les valeurs de E_{ref} et de la taille de la fenêtre d'entropie.

Nous focalisons l'analyse sur la situation la plus contraignante, correspondant à une taille de grains de 4096, à l'aide des données présentées dans la Figure 5.11. Bien que la qualité de l'adaptation reste sensible

Taille de grain	Capacité moyenne minimale		Capacité moyenne maximale	
	Seuil fixe	*Seuil dynamique*	*Seuil fixe*	*Seuil dynamique*
2048	1,996	1,996	2,001	2
3072	2,99	2,981	3,001	3
4096	3,984	3,962	4	4,001

TABLE 5.3 – Capacité moyenne atteinte par les couples $\{E_{ref}$; fenêtre d'entropie$\}$ selon la taille des grains injectés dans le tamis avec adaptation proportionnelle par entropie locale, avec et sans seuil critique dynamique. Pour chaque taille, les valeurs minimales et maximales des couples sont proposées.

aux paramètres, comme dans le cas de la méthode naïve, les écarts de performance entre les meilleurs et les pires couples sont nettement moindres ici. Cela traduit un meilleur contrôle des capacités de tamisage, indépendamment des réglages choisis. Ce comportement est particulièrement visible dans les mesures de capacité maximale atteinte : les pics restent systématiquement inférieurs à 15 pour la majorité des couples. À l'inverse, la méthode naïve dépasse souvent ce seuil. De plus, un plus grand nombre de couples est viable, et la majorité des couples restants ne provoque que peu d'avalanches infinies.



(a) Nombre d'avalanches infinies survenues.

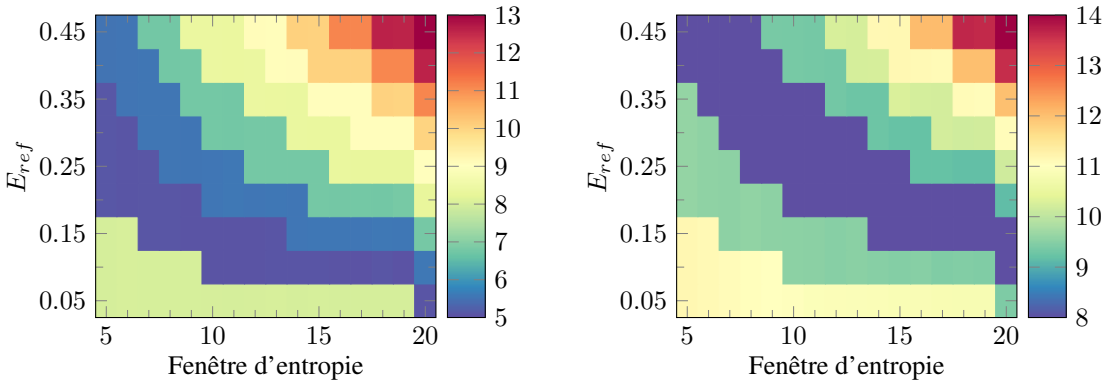
(b) Capacité maximale atteinte.

FIGURE 5.11 – Résultats de l'adaptation proportionnelle par entropie locale pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains à quatre fois la taille du système.

Malgré ces résultats globalement meilleurs, la méthode proportionnelle présente un désavantage lors de la sélection des paramètres optimaux : les capacités maximales atteintes sont directement corrélées aux paramètres, en particulier au produit $E_{ref} \times$ fenêtre. Prenons par exemple le couple $\{0{,}2\,;\,16\}$, l'un des plus performants. L'entropie locale nécessaire pour augmenter la capacité est ici de $\lceil 0{,}2 \times 16 \rceil = 4$, ce qui est précisément la capacité cible. Les cellules oscillent entre des capacités de 1 et 4, pour maintenir une entropie locale autour de E_{ref}. Si la tension s'accroît, davantage de grains sont observés, entraînant une augmentation proportionnelle de la capacité. Ainsi, le choix des paramètres optimaux dépend fortement de la charge injectée dans le système lorsque celle-ci est constante. La Figure 5.12 montre que les couples minimisant la capacité maximale sont ceux pour lesquels :

$$\lceil E_{ref} \times \text{fenêtre} \rceil = \frac{\text{taille des grains}}{\text{taille du système}}$$

Cette relation rend le dispositif expérimental inadapté à la sélection d'un couple générique de paramètres, car il est trop dépendant de la charge.



(a) Taille de grains : 2048.

(b) Taille de grains : 3072.

FIGURE 5.12 – Capacités maximales atteintes avec l'adaptation proportionnelle par entropie locale pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains de deux et trois fois la taille du système.

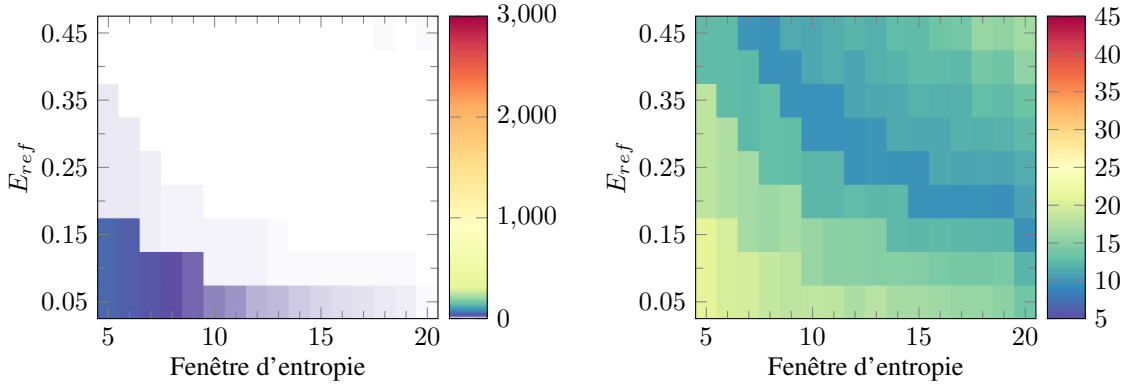
Pour dépasser cette limitation, nous avons placé le système dans une situation de charge fluctuante : la taille des grains injectés varie selon une sinusoïde. Les résultats, présentés dans la Figure 5.13, révèlent plusieurs tendances majeures :

— La structure des résultats est très proche de celle obtenue avec des grains de taille fixe de 4096.

— Peu d'avalanches infinies apparaissent, même chez les couples non optimaux.

— Avec l'introduction du seuil critique dynamique (Figure 5.14), presque tous les couples deviennent viables. Seuls quelques cas isolés présentent une unique avalanche infinie.

— Les capacités maximales atteintes sont globalement inférieures à celles obtenues sans le seuil dynamique.

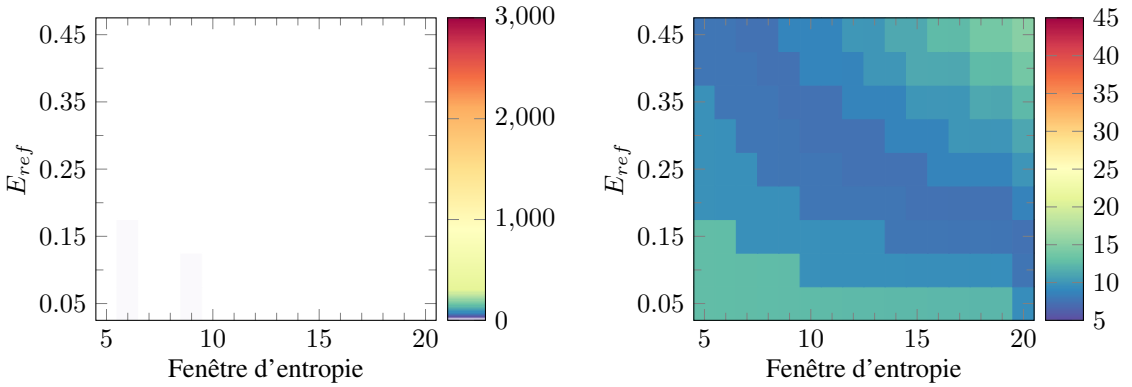


(a) Nombre d'avalanches infinies survenues.

(b) Capacité maximale atteinte.

FIGURE 5.13 – Résultats de l'adaptation proportionnelle par entropie locale pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains sinusoïdale.

Un autre point notable est que la consommation énergétique devient presque indépendante des paramètres, une fois le seuil dynamique activé (Figure 5.15). Comme pour la méthode naïve, les couples viables (ou quasi-

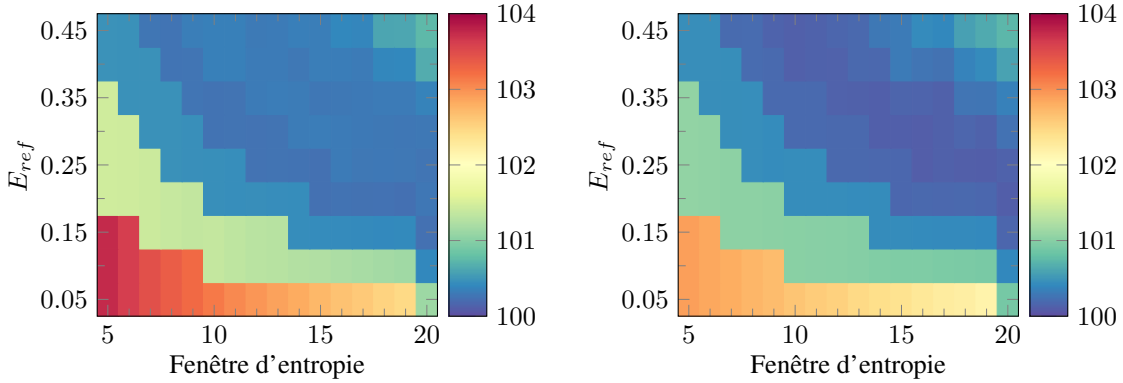


(a) Nombre d'avalanches infinies survenues.

(b) Capacité maximale atteinte.

FIGURE 5.14 – Résultats de l'adaptation proportionnelle par entropie locale avec seuil critique dynamique pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains sinusoïdale.

viables) maintiennent une consommation énergétique faible et stable, même sans ajustement dynamique du seuil. Cela suggère une capacité intrinsèque à équilibrer le traitement des grains excédentaires avec la tension nécessaire à l'auto-organisation, sans recours excessif aux ressources.



(a) Sans seuil dynamique.

(b) Avec seuil dynamique.

FIGURE 5.15 – Consommation énergétique du tamis utilisant l'adaptation proportionnelle par entropie locale, avec et sans seuil dynamique, pour chaque couple $\{E_{ref}$; fenêtre d'entropie$\}$ pour une taille de grains sinusoïdale.

En considérant l'ensemble des critères (viabilité, limitation des pics de capacité, faible consommation énergétique), le couple$\{0{,}25\,;\,14\}$ a été retenu comme référence pour la méthode proportionnelle. Ce couple est :

— est viable dans les scénarios avec ou sans seuil dynamique,

— présente l'une des capacités maximales les plus faibles,

— et produit l'une des consommations énergétiques les plus basses, y compris sous charge fluctuante.

Il sera utilisé pour comparer les deux approches d'adaptation par entropie locale dans la suite de l'étude.

5.4.5 Comparaison des méthodes

Bien que des indices aient déjà suggéré la supériorité de la méthode proportionnelle d'adaptation par entropie locale, nous procédons ici à une comparaison directe entre les deux approches. Les paramètres retenus sont {0,25 ; 11} pour la méthode naïve et {0,25 ; 14} pour la méthode proportionnelle, chacun représentant un couple optimal selon les analyses précédentes. Toutes les expériences ont été réalisées dans des conditions identiques à celles utilisées pour la sélection des paramètres.

Le Tableau 5.4 présente, pour différentes tailles de grains et scénarios de charge, la capacité maximale atteinte par les cellules ainsi que la consommation énergétique du système. Les résultats montrent que la méthode proportionnelle consomme légèrement moins d'énergie que la méthode naïve, avec une consommation proche de l'optimum, et maintient une capacité de tamisage maximale systématiquement deux à trois fois inférieure, ce qui traduit une adaptation plus fine et maîtrisée.

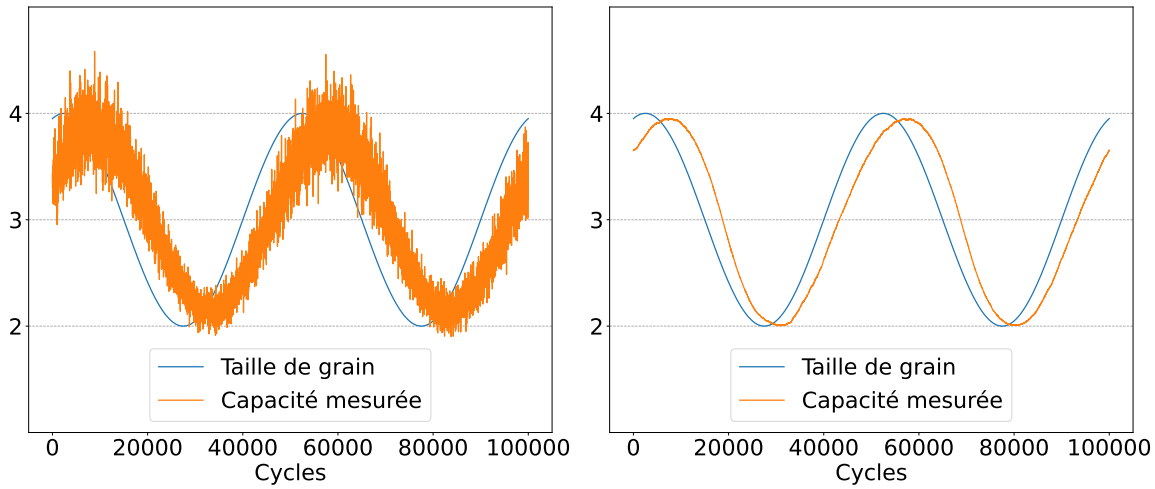
Taille de grain	Méthode	E_{ref}	Fenêtre d'entropie	Capacité max.	Consommation
2048	Naïve	0,25	11	16	102,06%
	Proportionnelle	0,25	14	**6**	**100,1%**
3072	Naïve	0,25	11	21	102,49%
	Proportionnelle	0,25	14	**8**	**100,1%**
4096	Naïve	0,25	11	26	102,55%
	Proportionnelle	0,25	14	**9**	**100,33%**
Sinusoïde	Naïve	0,25	11	23	102,63%
	Proportionnelle	0,25	14	**7**	**100,17%**
Aléatoire	Naïve	0,25	11	24	103,15%
	Proportionnelle	0,25	14	**8**	**101,18%**

TABLE 5.4 – Comparaison des couples de paramètres sélectionnés pour les méthodes d'adaptation par entropie locale pour les différentes tailles de grain.

Les Figures 5.16 et 5.17 illustrent l'évolution de la capacité moyenne du système en réponse à des charges dynamiques : une charge sinusoïdale (Figure 5.16), et une charge aléatoire comportant des pics de surcharge (Figure 5.17). Dans chaque graphique, la courbe bleue représente la taille du grain injecté (normalisée par rapport à la taille du système), tandis que la courbe orange montre la capacité moyenne de tamisage du système.

Les résultats mettent en évidence des différences notables. La méthode naïve génère une réponse instable et très fluctuante, ce qui reflète une forte sensibilité du système aux variations de charge. La méthode proportionnelle, en revanche, assure une adaptation progressive, lissée et robuste, même face à des charges extrêmes. En particulier, le scénario avec pics montre que le système proportionnel encaisse des surcharges importantes sans réaction excessive, ce qui témoigne d'un mécanisme de régulation efficace et durable.

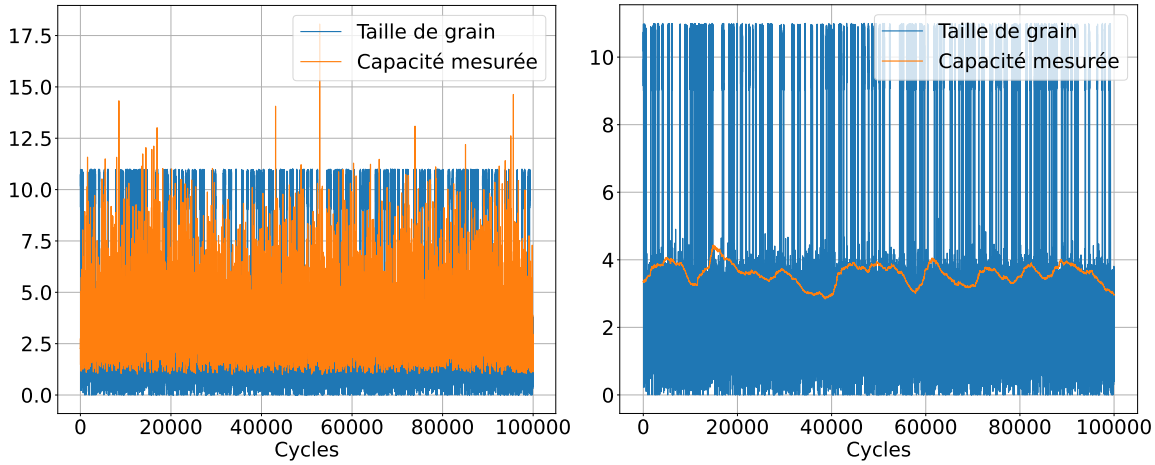
Sur l'ensemble des critères (capacité maximale, stabilité, consommation énergétique, et résilience face aux charges fluctuantes) la méthode proportionnelle s'avère nettement supérieure à la méthode naïve pour l'adaptation basée sur l'entropie locale.



(a) Méthode naïve.

(b) Méthode proportionnelle.

FIGURE 5.16 – Évolution de la capacité de tamisage d'un tamis de taille 32 pour les deux méthodes d'adaptation par entropie locale pour une taille de grain sinusoïdale. Les valeurs sont normalisées par rapport à la taille du système. Par exemple, une valeur de 3 correspond réellement à 3072.



(a) Méthode naïve.

(b) Méthode proportionnelle.

FIGURE 5.17 – Évolution de la capacité de tamisage moyenne des cellules d'un tamis de taille 32 pour les deux méthodes d'adaptation par entropie locale pour une taille de grain aléatoire avec des pics de grosse surcharge. Les valeurs sont normalisées par rapport à la taille du système.

5.5 Adaptation des capacités par protocole de bavardage

L'adaptation par protocole de bavardage constitue une nouvelle stratégie permettant d'ajuster dynamiquement les capacités de tamisage des cellules. À l'instar de la méthode fondée sur l'entropie locale, elle exploite le mouvement des grains comme indicateur d'activité, mais adopte une approche plus directe, sans mesure explicite ni traitement statistique. Cette stratégie repose sur la combinaison de deux mécanismes déjà présentés et introduit un troisième élément clé : la communication de l'état "bordure" entre cellules, rendue possible par la propagation d'avalanches locales.

Dans un premier temps, nous détaillons le fonctionnement de l'algorithme et ses mécanismes internes.

Nous analysons ensuite les dynamiques d'adaptation induites par cette approche, après avoir posé le cadre méthodologique de l'étude.

5.5.1 Modélisation

L'algorithme est initialisé en affectant aléatoirement l'état "bordure" à une faible portion des cellules du tamis. Ces cellules, considérées comme sources de l'état, conservent définitivement ce statut, même en l'absence de grains. Leur nombre doit rester limité afin d'éviter une surréaction du système au moindre mouvement.

Lors d'une avalanche, toute cellule recevant un grain depuis une voisine en état "bordure" adopte ce nouvel état et incrémente sa capacité de tamisage de 1, suivant un principe analogue au seuil critique dynamique. À l'instar de l'adaptation proportionnelle par entropie locale, une cellule vidée de ses grains réinitialise sa capacité à sa valeur minimale (1) et perd l'état "bordure", sauf si elle fait partie des cellules sources. Ainsi, les cellules sans état conservent un fonctionnement minimal, mais continuent à participer aux avalanches, contribuant pleinement à l'auto-organisation du système.

Ce mécanisme permet un engagement progressif des cellules dans l'adaptation en fonction de la tension présente dans le système. Plus les avalanches sont fréquentes, plus l'état "bordure" se propage, et plus les capacités de tamisage augmentent collectivement. Inversement, la perte de l'état "bordure" permet de désengager les cellules lorsque la charge diminue, maintenant ainsi un régime général de tamisage adapté à la demande. À l'échelle d'une cellule, la capacité de tamisage oscille dynamiquement selon l'activité locale, produisant une capacité moyenne globale proportionnelle à l'intensité du flux de grains injectés.

Contrairement aux méthodes par entropie locale, l'adaptation par protocole de bavardage n'introduit pas de phase dédiée dans le cycle de simulation. Le mécanisme est intégré directement à la gestion des avalanches, à la manière du seuil critique dynamique. L'Algorithme 9 en présente le pseudo-code détaillé.

La Figure 5.18 illustre la distribution spatio-temporelle de l'état "bordure" dans un tamis de taille 32, selon différentes tailles de grains :

— À 50% de la capacité du tamis (512), la tension est quasi inexistante, les avalanches rares, et seules quelques cellules proches des sources adoptent brièvement l'état.
— À 100% de la capacité (1024), la tension devient significative : l'état "bordure" est transmis à l'ensemble du tamis, bien que principalement autour des cellules sources.
— Pour une charge à 200% (2048), la propagation est massive : la majorité des cellules sont "bordure" environ la moitié du temps.
— Enfin, à 400% (4096), les avalanches deviennent omniprésentes et la quasi-totalité des cellules sont en état "bordure" quasi permanent.

L'évolution de la présence de l'état "bordure" reflète donc la réponse adaptative du système. Étant donné que chaque acquisition de cet état s'accompagne d'une augmentation de la capacité de tamisage, cette dynamique de propagation implique l'augmentation des capacités des capacités de tamisage des cellules.

Algorithm 9: Auto-adaptation par protocole de bavardage : incorporation dans les avalanches

Input: G : grille

```
   /* Gestion d'une avalanche                                          */
1  while au moins une cellule de G est instable do
2      foreach cellule de G do
           /* Éboulement de la cellule                                 */
3          if cellule.grains ≥ 4 then
4              foreach voisine de cellule do
5                  voisine.grains ← voisine.grains + 1
6
                   /* Communication de l'état bordure et incrémentation de la
                      capacité                                         */
7                  if cellule est "bordure" then
8                      Communication de l'état "bordure" à voisine
9                      voisine.capacité = voisine.capacité + 1
10                 end
11             end
12             cellule.grains ← cellule.grains − 4
13         end
14     end
15 end
```

5.5.2 Cadre d'étude

Le cadre d'étude pour l'évaluation de l'adaptation par protocole de bavardage est calqué sur celui défini pour l'adaptation par entropie locale (voir Section 5.4.4.1). Cependant, le panel de tailles de grain des les politiques fixes est élargi, s'étendant de 256 à 4096, afin d'explorer des régimes de sous-charge. En outre, l'initialisation du protocole de bavardage est réalisée avec seulement 1% des cellules définies comme sources de l'état "bordure", afin d'introduire une hétérogénéité minimale dans le système initial.

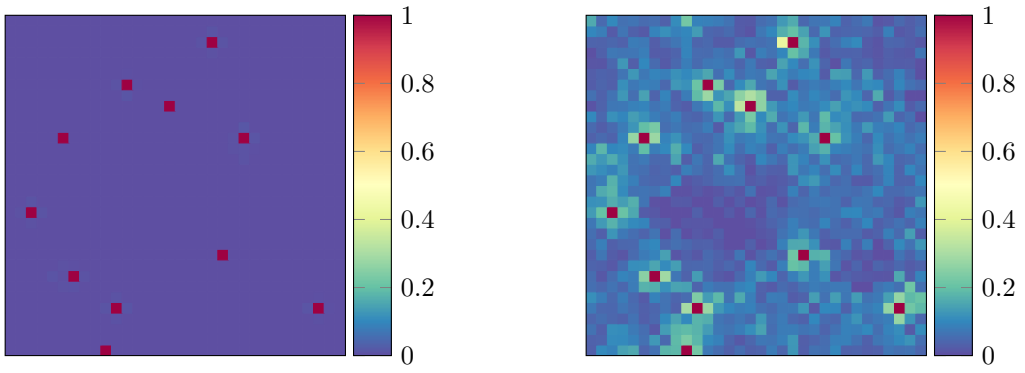
Les métriques d'analyse utilisées pour évaluer l'efficacité de cette stratégie sont identiques à celles employées pour l'adaptation par entropie locale. Deux métriques complémentaires sont toutefois introduites pour caractériser plus finement le comportement du système :

— **Densité :** mesure le nombre moyen de grains présents sur les cellules, normalisée par le seuil critique d'éboulement. Une densité égale à 1 signifie que toutes les cellules sont à un grain de s'ébouler, représentant un état de tension maximale où le système est plein.

— **Taux d'utilisation des cellules :** correspond au rapport entre le nombre de cycles pendant lesquels une cellule traite un grain et la durée totale de la simulation. Un taux égal à 1 indique que la cellule a été sollicitée en continu, ce qui reflète un niveau élevé d'activité.

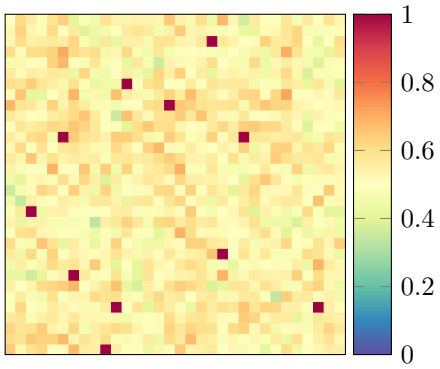
5.5.3 Analyse de l'adaptation

À l'instar de l'adaptation par entropie locale, l'adaptation par protocole de bavardage s'avère pleinement fonctionnelle. Le Tableau 5.5 présente les résultats obtenus pour plusieurs scénarios de charge, en termes de capacités moyennes et maximales, consommation énergétique et nombre d'avalanches infinies. Les résultats

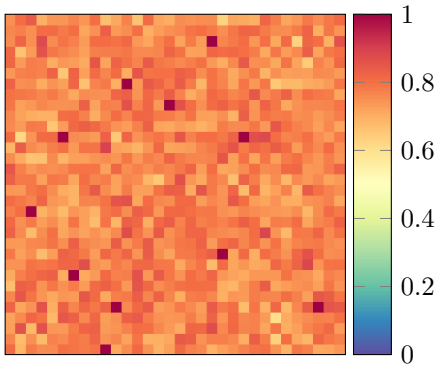


(a) Taille de grain : 512.

(b) Taille de grain : 1024.



(c) Taille de grain : 2048.



(d) Taille de grain : 4096.

FIGURE 5.18 – Taux de présence de l'état "bordure" sur les cellules d'un tamis de taille 32 utilisant l'adaptation par protocole de bavardage pour différentes tailles de grain. Le taux correspond au ratio entre le nombre de cycles où chaque cellule est en état "bordure" par rapport à la durée totale de la simulation. Une valeur à 1 signifie que la cellule a été "bordure" durant toute la simulation. C'est notamment le cas des cellules sources de l'état (1% des cellules) en rouge foncé.

montrent que les capacités de tamisage s'ajustent de manière efficace, avec des valeurs maximales modérées, une consommation énergétique proche de l'optimal, et aucune avalanche infinie, quel que soit le scénario considéré.

Taille de grain	Capacité moyenne	Capacité maximale	Avalanches infinies	Consommation
2048	2	8	0	100,36%
3072	3	11	0	100,15%
4096	3,99	14	0	99,81%
Sinusoïdale	3	11	0	100,15%
Aléatoire	3,59	14	0	101,14%

TABLE 5.5 – Résultats de l'adaptation par protocole de bavardage dans un tamis de taille 32. Chaque ligne correspond à un scénario de charge. Les trois premiers sont des tailles de grain fixes de 2048, 3072 et 4096. Le dernier est une taille fluctuante qui suit une sinusoïde oscillant entre 2048 et 4096 sur une période de 50000 cycles.

La réactivité élevée de cette méthode permet une augmentation rapide des capacités locales, ce qui réduit fortement la tension dans le système, en comparaison au tas de sable canonique ou aux modèles d'adaptation précédents. Comme le montre le Tableau 5.6, lorsque que la taille des grains injectés est supérieure ou égale

à la taille du système, la tension reste faible, le système n'étant même pas rempli à moitié. Cela se traduit par une réduction significative du nombre et de la durée des avalanches, qui ne dépassent plus dix étapes. Ainsi, les redistributions massives de grains sont évitées, tout comme les avalanches infinies, rendant inutile l'usage du seuil critique dynamique.

Taille de grain	256	512	1024	2048	4096
Densité	0,01	0,07	0,41	0,34	0,40
Cellules "bordures" 50% du temps	0	0	0	95%	100%

TABLE 5.6 – Densité de grains et taux de cellules ayant l'état "bordure" au moins 50% du temps dans un tamis de taille 32 utilisant l'adaptation par protocole de bavardage pour différentes tailles de grain. Une densité de 1 signifie que toutes les cellules sont à 1 grain de s'ébouler, tandis que 0 signifie que le système est vide.

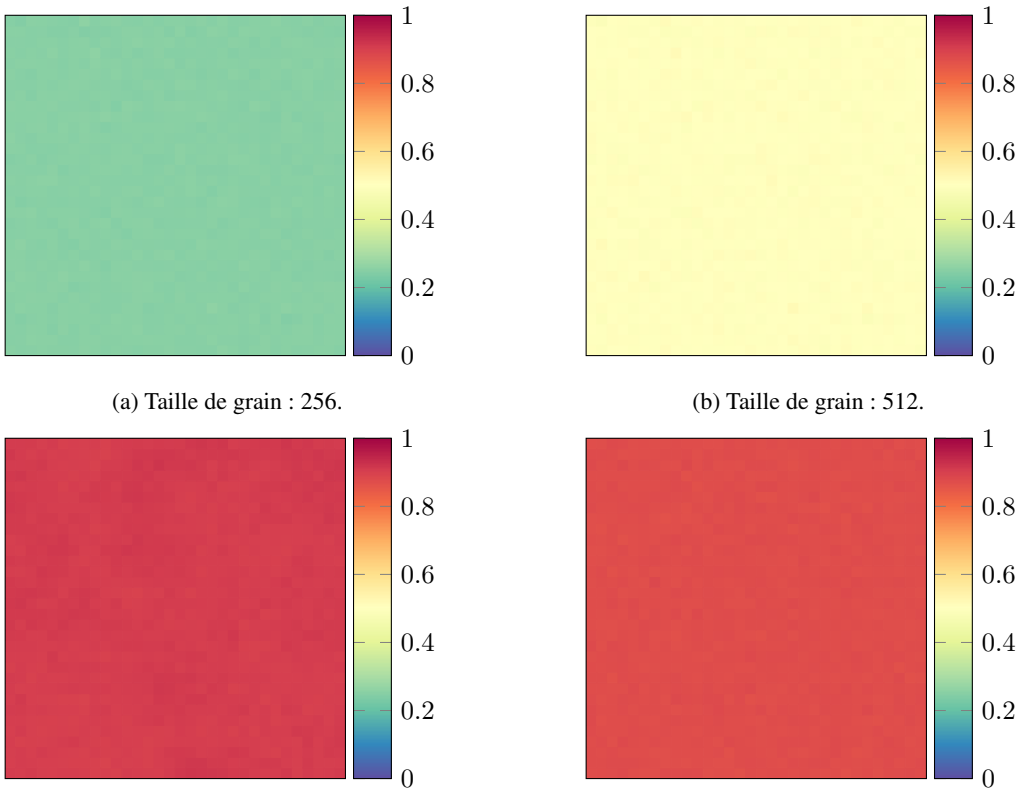
Dans les scénarios de sous-charge (taille de grain inférieure à 1024), la tension est quasi nulle, les cellules ayant le temps de traiter chaque grain avant d'en recevoir un autre. Le pic de densité est atteint à 1024, ce qui correspond exactement à la capacité de traitement du système en un cycle. À cette valeur, le système opère à la limite entre sous-charge et surcharge, générant très peu d'avalanches. Dans cette configuration, l'adaptation des capacités ne s'active que marginalement, comme en témoigne un taux nul de cellules en état "bordure" au moins 50% du temps. Cela suffit néanmoins à préserver le système d'une surcharge.

À partir d'une taille de grain de 2048, l'adaptation s'active de manière globale, presque toutes les cellules adoptant une capacité accrue au moins la moitié du temps. Ce déclenchement global, bien que modeste (augmentation de capacité d'une unité seulement), réduit la tension en permettant à certaines cellules de surpasser temporairement les besoins, tandis que d'autres demeurent à un niveau minimal d'activité. Pour 4096, une taille de grain plus élevée, davantage de mouvement est requis pour atteindre les capacités de tamisage nécessaires, ce qui entraîne une reprise de l'augmentation de la densité.

Malgré une tension généralement faible, les petites avalanches locales permettent une redistribution des grains, assurant une utilisation quasi complète du système dès que la taille des grains est au moins égale à celle du système. Comme le montre la Figure 5.19, le taux d'utilisation des cellules reste élevé pour ces cas, tandis qu'en sous-charge, il est proportionnel à la charge injectée, soit $\frac{256}{1024} = 25\%$ et $\frac{512}{1024} = 50\%$.

L'adaptation par protocole de bavardage montre également une bonne capacité à suivre des charges dynamiques. La Figure 5.20a illustre l'évolution de la capacité de tamisage moyenne face à une charge sinusoïdale, démontrant une adaptation continue et fluide à la variation de la taille des grains. On observe cependant une phase transitoire amortie d'environ 10 000 cycles, caractéristique de cette méthode, correspondant à une oscillation autour de la valeur cible avant stabilisation. Ce phénomène est systématiquement observé, indiquant que le temps d'initialisation de cette approche est plus long que celui des méthodes basées sur l'entropie locale.

Enfin, la Figure 5.20b met en évidence la résilience du protocole de bavardage face aux charges aléatoires et aux pics inattendus. Bien que la capacité de tamisage moyenne oscille en réponse aux variations, cette oscillation reste modérée et centrée autour de la capacité optimale (environ 3,55), garantissant ainsi une réactivité efficace sans excès.



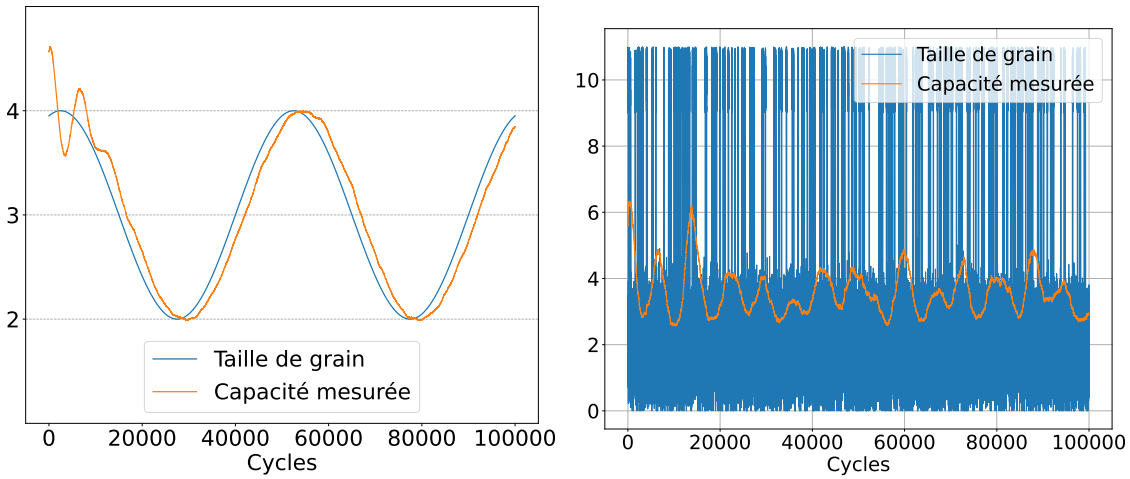
(a) Taille de grain : 256.

(b) Taille de grain : 512.

(c) Taille de grain : 1024.

(d) Taille de grain : 2048.

FIGURE 5.19 – Taux d'utilisation des cellules d'un tamis de taille 32 utilisant l'adaptation par protocole de bavardage pour différentes tailles de grain. Le taux correspond au ratio entre le nombre de cycles où chaque cellule traite un grain par rapport à la durée totale de la simulation. Une valeur à 1 signifie que la cellule a été utilisée durant toute la simulation.



(a) Charge sinusoïdale.

(b) Charge aléatoire avec pics.

FIGURE 5.20 – Évolution de la capacité moyenne de tamisage dans un tamis de taille 32 utilisant l'adaptation par protocole de bavardage. La taille des grains, relative à la taille du système, est sinusoïdale et aléatoire.

5.6 Comparaison des méthodes

Dans cette section, nous allons comparer les deux stratégies d'adaptation des capacités : l'adaptation proportionnelle par entropie locale couplée au seuil dynamique, et l'adaptation par protocole de bavardage. Le cadre d'étude sera d'abord posé. Puis nous comparerons les deux méthodes pour les scénarios de charge que nous avons déjà étudié pour les méthodes séparées : différentes charges fixes et charge fluctuante. Nous nous intéresserons ensuite aux performances des méthodes pour un scénario de charge réelle. Enfin, nous discuterons des avantages et des inconvénients de chaque méthode.

5.6.1 Cadre d'étude

Le cadre d'étude pour comparer les deux approches d'adaptation des capacités de tamisage du tamis se divise en deux. La première partie, très similaire à ce que nous avons déjà vu précédemment, concerne les scénarios de charge fixes et fluctuantes pour lesquels le système est constamment surchargé. La seconde partie concerne le cas du scénario de charge réelle.

5.6.1.1 Charge fixe et charge fluctuante

Pour les scénarios de charge fixe et fluctuante, le cadre d'étude demeure identique à celui employé dans l'étude individuelle des deux méthodes d'adaptation. Cependant, de nouvelles métriques viennent compléter celles utilisées précédemment, et la consommation énergétique ne correspond désormais plus qu'aux avalanches, tout en étant exprimée en ratio par rapport à la consommation totale du système. Les nouvelles métriques sont les suivantes :

— **Consommation énergétique des avalanches :** cumul de tous les mouvements de grains pendant les avalanches. Cette consommation est exprimée en ratio par rapport à la consommation totale (tamisage et avalanche).

— **Indice de Jain :** mesure de l'égalité de la répartition des grains sur les cellules. Nous utilisons ici la version simple de l'indice (Équation 2.1), adaptée au contexte dans lequel les capacités de tamisage sont en constante évolution et où la file d'attente de grains est courte. Bien que le tamis auto-adaptatif soit conçu pour gérer la surcharge plutôt que pour garantir un équilibrage parfait de la charge, cette métrique fournit une indication sur l'état du système après la surcharge. Une valeur de 1 indique une répartition parfaite.

— **Débit :** moyenne du nombre de grains tamisés à chaque cycle de simulation. Une valeur à 1 indique qu'en moyenne, un grain est traité et retiré du système à chaque cycle.

— **Durée de vie d'un grain :** nombre de cycles qu'un grain passe dans le tamis, entre son introduction dans le système et son tamisage complet. Cette métrique reflète le temps de réponse du système, notion évoquée au Chapitre 2, et peut être interprétée comme un indicateur de qualité de service.

Enfin, les paramètres spécifiques à chaque méthode sont les suivants :

— l'adaptation par entropie locale utilise le couple de paramètres $\{0{,}25\,;\,14\}$;

— l'adaptation par protocole de bavardage est initialisée avec 1% de cellules sources de l'état "bordure".

5.6.1.2 Charge réelle

La charge réelle utilisée dans ce scénario est extraite du projet Grid Workload Archive IOSUP et al. (2008a), qui met à disposition les traces d'exécution de divers systèmes de calcul distribués. Les détails relatifs à la provenance de ces données, ainsi que leur format, sont présentés en Section B.2.2 de l'Annexe B, dédiée à la reproductibilité des expériences. Dans le cadre de ce travail, nous exploitons plus précisément les traces des systèmes AuverGrid, NorduGrid et SHARCNET.

Étant donné la longueur totale des traces (allant de plusieurs mois à plusieurs années, donc des dizaines de millions de secondes), nous restreignons notre étude à une fenêtre de deux semaines d'activité, soit 1 209 600 secondes (cycles). Contrairement aux précédentes études où un seul grain était injecté par cycle, plusieurs tâches peuvent ici être soumises simultanément, ce qui rapproche ce scénario des modèles de type sac-de-tâches abordés en Section 3.5.1.

Afin de mettre les mécanismes d'adaptation à l'épreuve, la taille du tamis est volontairement réduite à 100 cellules (grille toroïdale de 10×10), soit un nombre de ressources bien inférieur à celui des centres de calcul étudiés (475, 2000 et 6828). Cela est justifié par l'observation que ces systèmes sont rarement utilisés à pleine capacité, comme le montrent les analyses statistiques accompagnant les traces.

Les simulations s'exécutent jusqu'à ce que le tamis soit vide, après l'injection des grains correspondant aux tâches des 1 209 600 secondes à partir des dates de début suivantes :

— **AuverGrid** : 1er avril 2006 ;
— **NorduGrid** : 1er janvier 2006 ;
— **SHARCNET** : 1er septembre 2006.

Ces dates correspondent à des périodes durant lesquelles de nombreuses tâches ont été soumises aux systèmes.

L'objectif de ce scénario est d'évaluer la capacité du tamis auto-adaptatif à gérer une charge réelle, selon les deux méthodes d'adaptation des capacités de tamisage. Pour ce faire, le fonctionnement idéal des centres est simulé à partir des traces, en considérant que chaque tâche est affectée à un processeur unique de capacité 1, lequel disparaît une fois la tâche terminée. Ce fonctionnement de référence permet d'évaluer les performances du tamis adaptatif selon les métriques suivantes :

— **Temps total d'exécution (*makespan*) :** durée nécessaire pour tamiser l'ensemble des grains insérés, jusqu'à ce que le système soit vide.
— **Consommation d'énergie :** somme des capacités effectives utilisées et des déplacements de grains, exprimée en pourcentage de la consommation de référence dérivée des traces.
— **Capacité de tamisage globale :** capacité de tamisage totale (somme des capacités effectives) de toutes les cellules à chaque cycle, représentant la "puissance instantanée" du système.

Les paramètres restent identiques aux études précédentes : $\{0{,}25\,;\,14\}$ pour l'adaptation par entropie locale, et une seule cellule (soit 1%) initialement en état "bordure" pour le protocole de bavardage.

Enfin, la nature des charges (potentiellement plusieurs grains injectés à chaque cycle) rend nécessaire le

couplage systématique des deux approches avec le seuil critique dynamique. Celui-ci permet d'amortir les pics de soumission simultanée et d'éviter les avalanches prolongées lorsque le système est temporairement saturé. Dans ce cadre, les avalanches ne sont plus limitées à une durée maximale de 500 itérations.

5.6.2 Scénarios de charge fixe et fluctuante

Les résultats sont synthétisés dans les tableaux complémentaires 5.7 et 5.8. Le premier tableau rassemble les mesures pour la phase opérationnelle des simulations (100 000 cycles après initialisation ; 1 grain est injecté à chaque cycle) : la capacité de tamisage moyenne et maximale, la consommation d'énergie liée aux avalanches et l'indice de Jain. Le second tableau, pour une simulation complète (phase d'initialisation de 10 000 cycles ; phase opérationnelle de 100 000 cycles avec injection de grains, et phase de vidange complète du système), ne propose plus la capacité moyenne et l'indice de Jain qui deviennent des métriques non pertinentes à cause des phases d'initialisation et de vidange. En revanche, le débit est ajouté.

Comme nous l'avons déjà observé dans les études précédentes menées de manière individuelle, les deux méthodes d'adaptation des capacités de tamisage sont fonctionnelles et permettent au système de répondre efficacement à la charge. Néanmoins, quelques différences apparaissent lorsqu'elles sont mises en comparaison directe, comme le montrent les résultats synthétisés dans le Tableau 5.7.

Taille de grain	**Méthode**	**Capacité**		**Consommation des avalanches**	**Indice de Jain**
		Moyenne	*Maximale*		
2048	Entropie locale	2	6	0,21%	0,7
	Prot. de bavardage	2	8	0,04%	0,51
3072	Entropie locale	3	8	0,2%	0,72
	Prot. de bavardage	3	11	0,03%	0,55
4096	Entropie locale	4	9	0,31%	0,72
	Prot. de bavardage	3,99	14	0,03%	0,58
Sinusoïde	Entropie locale	3	7	0,23%	0,71
	Prot. de bavardage	3	11	0,03%	0,54
Aléatoire	Entropie locale	3,58	8	0,25%	0,71
	Prot. de bavardage	3,59	14	0,03%	0,54

TABLE 5.7 – Comparaison des méthodes d'auto-adaptation selon la tailles des grains injectés dans le tamis. Les résultats concernent les 100 000 cycles de simulation après initialisation, pour lesquels un grain est injecté dans le tamis à chaque cycle. La capacité de tamisage moyenne doit être au plus proche de $\frac{\text{taille des grains}}{\text{taille du système}}$, soit 2, 3, 4, 3 et 3,55 pour les tailles proposées. Une capacité de tamisage maximale faible indique une adaptation plus homogène et contrôlée. La consommation des avalanches représente la quantité d'énergie dépensée par les avalanches par rapport à la consommation totale (avalanches et tamisage). Plus l'indice de Jain est proche de 1, plus l'équilibrage des grains sur les cellules est intéressant.

Sur le plan des capacités de tamisage, les deux approches atteignent des valeurs moyennes très similaires et nécessaires, quels que soient les scénarios. Cependant, l'adaptation par protocole de bavardage tend à pousser localement les capacités plus haut, ce qui entraîne un tamisage hétérogène ; c'est-à-dire que certaines zones du système sont en sur-tamisage tandis que d'autre sont en sous-tamisage par rapport à l'objectif, bien qu'à l'échelle du système, la capacité de tamisage soit au nécessaire. À l'inverse, l'adaptation par entropie locale a

un meilleur contrôle de l'ajustement des capacités.

Cette dynamique a des conséquences sur les mécanismes d'organisation du système. La plus grande réactivité du protocole de bavardage impacte significativement la tension interne, limitant les déplacements de grains et donc les avalanches. Par conséquent, la consommation énergétique liée aux avalanches est nettement plus faible avec cette méthode qu'avec celle basée sur l'entropie locale. Cela dit, dans les deux cas, cette composante énergétique demeure très marginale comparée à celle induite par le tamisage lui-même. La consommation énergétique totale, majoritairement définie par le tamisage, reste ainsi très proche du nécessaire pour les deux stratégies, avec des différences négligeables.

En ce qui concerne l'équilibrage de la charge, mesuré à l'aide de l'indice de Jain, l'adaptation par entropie locale s'avère plus efficace. La tension interne maintenue dans le système favorise une répartition plus homogène des grains à travers les cellules. À l'inverse, la méthode par protocole de bavardage, en réduisant fortement les avalanches, aboutit à une répartition de la charge plus locale et fragmentée, particulièrement entre zones éloignées. Cela ne pose pas de problème dans un contexte de surcharge constante, où la majorité des ressources sont sollicitées. En revanche, dans un scénario à fortes fluctuations, avec alternance de pics et de phases de sous-charge, cette fragmentation peut entraîner une sous-utilisation partielle du système.

Cette hétérogénéité, tant au niveau des capacités de tamisage que du niveau de charge, permet au protocole de bavardage de bénéficier d'un temps d'exécution total réduit. Il en résulte un débit légèrement supérieur à celui obtenu avec l'approche par entropie locale, comme le montre le Tableau 5.8. Les débits mesurés restent inférieurs à 1, dans la mesure où l'entièreté de la simulation (incluant les phases d'initialisation et de vidange) est prise en compte.

Taille de grain	Méthode	Capacité maximale	Consommation des avalanches	Débit
2048	Entropie locale	6	0,2%	0,88
	Prot. de bavardage	7	0,04%	0,93
3072	Entropie locale	6	0,2%	0,83
	Prot. de bavardage	10	0,03%	0,9
4096	Entropie locale	7	0,3%	0,79
	Prot. de bavardage	13	0,03%	0,88
Sinusoïde	Entropie locale	6	0,23%	0,71
	Prot. de bavardage	10	0,03%	0,79
Aléatoire	Entropie locale	7	0,24%	0,75
	Prot. de bavardage	12	0,03%	0,82

TABLE 5.8 – Comparaison des méthodes d'auto-adaptation selon la tailles des grains injectés dans le tamis. Les résultats concernent les simulations complètes : phase d'initialisation (10 000 cycles), phase opérationnelle avec injection de grains (100 000 cycles), et phase de vidange complète du système. Une capacité de tamisage maximale faible indique une adaptation plus homogène et contrôlée. La consommation des avalanches représente la quantité d'énergie dépensée par les avalanches par rapport à la consommation totale (avalanches et tamisage). Le débit correspond au nombre moyen de grain tamisé (sortant du système) à chaque cycle. Un débit élevé indique un tamisage plus rapide.

Par ailleurs, la mesure des capacités de tamisage maximales sur l'entièreté de la simulation présente une

baisse des valeurs. Cela est dû à l'inactivité majoritaire des cellules pendant l'initialisation et à leur inactivité progressive durant la vidange. La consommation associées aux avalanches reste toutefois pratiquement inchangée et demeure très marginale, malgré la présence de la phase d'initialisation qui tend à faire surréagir le système avant que l'adaptation ne le stabilise, en particulier pour le protocole de bavardage.

Enfin, du point de vue de la qualité de service, mesurée par la durée de vie des grains (cf. Figure 5.21), des différences significatives apparaissent. Dans les deux approches, aucun ordonnancement n'est appliqué localement pour les opérations de tamisage, et le déplacement des grains repose sur une dynamique de type marche aléatoire. Dans ce cadre, la tension plus forte induite par l'entropie locale, qui favorise un bon équilibrage, a ici un effet négatif : les grains peuvent être repoussés et circuler longtemps dans le tamis avant d'atteindre une cellule prête à les traiter, ce qui augmente leur durée de vie. À l'inverse, dans le protocole de bavardage, les grains se déplacent peu et attendent généralement peu (équivalent du tamisage d'un à trois grains) avant d'être tamisés. La qualité de service est ainsi nettement supérieure avec cette méthode, les grains étant traités de manière plus fluide et rapide.

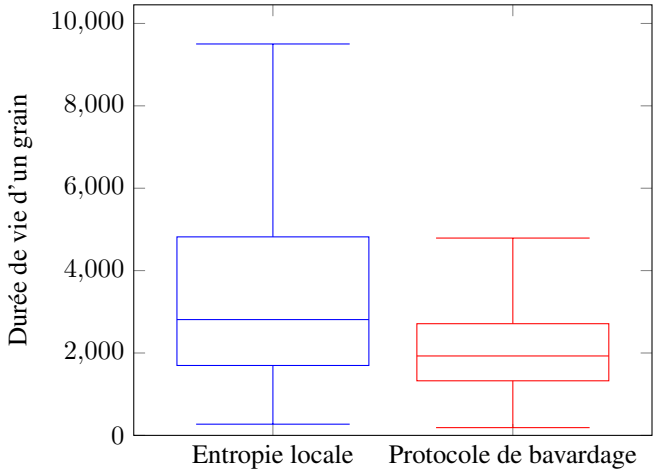


FIGURE 5.21 – Durée de vie des grains dans un tamis de taille 32 selon chaque méthode d'adaptation des capacités de tamisage pour la charge sinusoïdale. Chacune présente des valeurs aberrantes non présentes sur le graphique pour la lisibilité : jusqu'à 35000 pour la première, et jusqu'à 15000 pour la seconde.

D'après cette étude, le choix de la meilleure modélisation de l'adaptation des capacités de tamisage dépend des objectifs. Si l'on souhaite de la parcimonie dans l'utilisation des ressources, l'adaptation par entropie locale offre de meilleurs résultats. En revanche, si l'on s'intéresse plutôt à la qualité de service, l'adaptation par protocole de bavardage propose de bien meilleures performances.

5.6.3 Scénario de charge réelle

Comme le montre le Tableau 5.9, les deux méthodes d'adaptation des capacités de tamisage sont capables de faire face efficacement à des charges issues de systèmes réels, avec une consommation énergétique globalement équivalente à celle dérivée des systèmes de calcul dont proviennent les traces.

Sur la trace AuverGrid, l'approche par protocole de bavardage parvient à reproduire un temps total d'exécution très proche de la référence, voire légèrement inférieur. L'adaptation par entropie locale, en revanche,

Trace	Système	Temps d'exécution	Consommation des avalanches
AuverGrid	Référence	1 418 433	0%
	Entropie locale	1 565 105	0,013%
	Prot. de bavardage	**1 407 797**	**0,003%**
NorduGrid	Référence	6 371 445	0%
	Entropie locale	2 099 109	0,294%
	Prot. de bavardage	**1 407 116**	**0,013%**
SHARCNET	Référence	4 711 420	0%
	Entropie locale	4 899 089	0,171%
	Prot. de bavardage	**1 920 970**	**0,021%**

TABLE 5.9 – Résultats des méthodes d'auto-adaptation pour les traces AuverGrid, NorduGrid et SHARCNET.

nécessite environ 10% de cycles supplémentaires pour parvenir au traitement complet de la charge. Cela dit, cette différence n'impacte pas significativement la consommation d'énergie, les deux approches maintenant une dépense énergétique quasiment identique à celle des systèmes de référence. La Figure 5.22 illustre la réaction des modèles pour cette trace. On observe que pour le tamis auto-adaptatif, peu importe la méthode d'adaptation, il ajuste organiquement sa capacité de tamisage globale pour suivre les pics de charge.

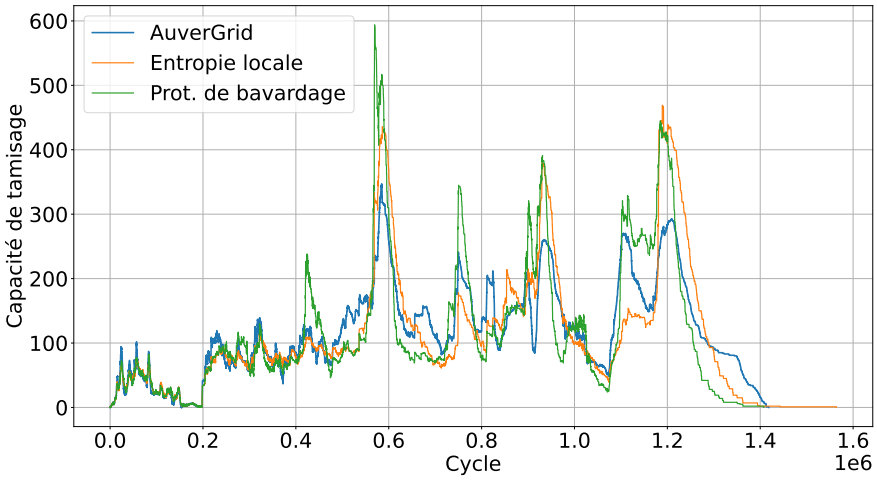


FIGURE 5.22 – Évolution de la capacité de tamisage globale des systèmes pour la trace d'AuverGrid.

Les traces NorduGrid et SHARCNET présentent une dynamique de soumission de tâches différente. La quantité de charge injectée est généralement très faible, interrompue par des pics soudains et brefs, contrairement à la trace AuverGrid qui présente une charge plus continue ponctuée de hausses modérées. De plus, des durées de tâches beaucoup plus longues sont présentes. Cette dynamique a pour effet d'activer violemment les mécanismes d'adaptation, exploitant la tension élevée induite par ces pics pour augmenter rapidement les capacités de tamisage. Le tamis parvient alors à traiter les grains beaucoup plus rapidement que les systèmes d'origine, tout en revenant rapidement à un état de veille, ce qui explique que la consommation énergétique totale demeure très proche de celle de référence.

Les Figures 5.23 et 5.24 illustrent l'évolution de la capacité de tamisage du système pour les traces Nordu-

Grid et SHARCNET. Les cycles au-delà de 2 millions ont été tronqués, car ils n'apportent pas d'information significative : la capacité y décroît progressivement. Par ailleurs, l'axe des ordonnées est représenté en échelle logarithmique afin de faciliter la lecture, rendue difficile par les réactions violentes observées ponctuellement.

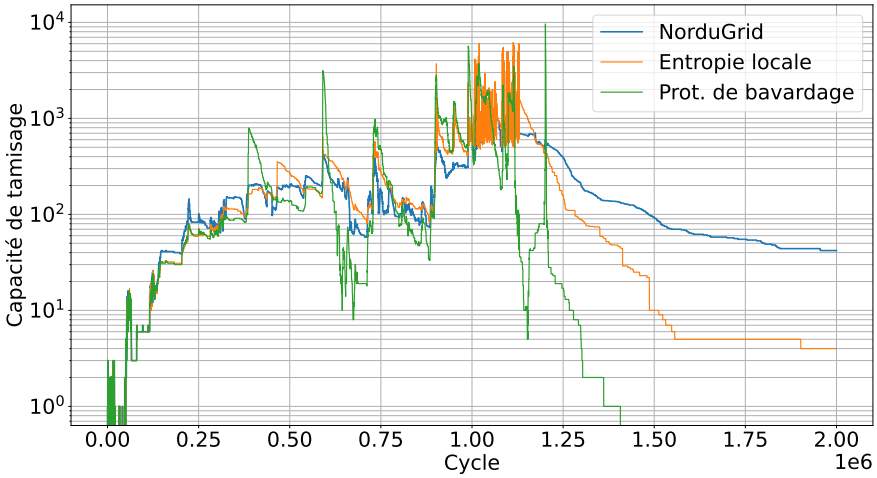


FIGURE 5.23 – Évolution de la capacité de tamisage globale des systèmes pour la trace de NorduGrid. La capacité de tamisage est affichée en échelle logarithmique.

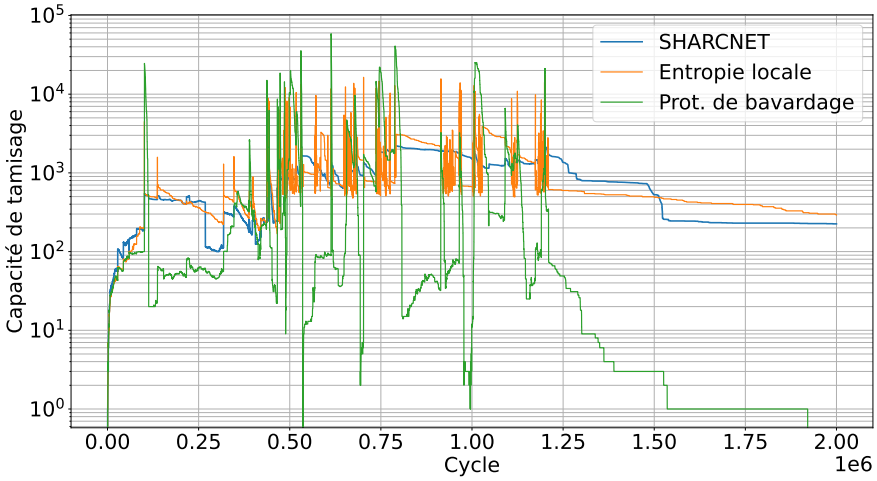


FIGURE 5.24 – Évolution de la capacité de tamisage globale des systèmes pour la trace de SHARCNET. La capacité de tamisage est affichée en échelle logarithmique.

Lorsque la capacité se stabilise autour de 100 (c'est-à-dire en l'absence ou en présence d'une adaptation minimale), le tamis auto-adaptatif réagit de manière satisfaisante. En revanche, les pics de charge soudains (visibles dans les données de référence par des sauts brusques) forcent le système à ajuster ses capacités, ce qui provoque des réactions particulièrement marquées. C'est notamment le cas pour l'adaptation par protocole de bavardage, qui atteint une capacité globale de 60 000 (soit 600 par cellule) sur la trace SHARCNET. Ces réactions brutales entraînent une vidange rapide du tamis, suivie d'une chute abrupte des capacités avant l'apparition d'un nouveau pic. La méthode basée sur l'entropie locale, bien qu'elle génère également des pics d'adaptation importants, produit des ajustements plus modérés. Cette modération se traduit notamment par un

temps de traitement légèrement plus élevé que la référence sur la trace SHARCNET.

L'adaptation plus limitée produite par l'entropie locale s'explique par les caractéristiques intrinsèques de la méthode : la mesure de l'entropie étant effectuée sur une fenêtre temporelle courte, la réaction adaptative s'affaiblit rapidement une fois le pic de charge passé. Une fois stabilisé par le seuil critique dynamique, le système présente une capacité de tamisage réduite, insuffisante pour achever rapidement le traitement des grains restants. À l'inverse, le protocole de bavardage conserve les capacités élevées atteintes pendant le pic, permettant de maintenir un régime de traitement plus intense jusqu'à complète vidange du tamis. En outre, dans un tel contexte, les performances de la méthode par entropie locale dépendent totalement de ses paramètres puisqu'il définissent la capacité d'adaptation minimale ($\lceil E_{ref} \times \text{fenêtre} \rceil$).

Les surréactions brèves mais intenses soulignent un aspect fondamental du décalage entre la conception du tamis auto-adaptatif et les pratiques des systèmes réels. Alors que le tamis est pensé pour s'adapter à une surcharge constante, les systèmes de calcul sont, eux, dimensionnés pour fonctionner en sous-charge, afin de pouvoir absorber des pics de demande. Lorsque le nombre de cellules du tamis est équivalent au nombre de processeurs des systèmes de référence, aucune avalanche ni adaptation n'est observée, les ressources étant suffisantes pour absorber la charge.

Les systèmes réels ayant une puissance limitée par le matériel, nous avons introduit une limitation arbitraire de la capacité de tamisage par cellule, fixée à 20, soit 2000 pour le système complet afin d'aller un peu plus loin. Le Tableau 5.10 présente les résultats obtenus pour NorduGrid et SHARCNET, la trace AuverGrid n'étant pas affectée par cette contrainte.

Trace	Système	Temps d'exécution	Consommation des avalanches
NorduGrid	Référence	6 371 445	0%
	Entropie locale	2 275 380	0,313%
	Prot. de bavardage	**1 261 919**	**0,031%**
SHARCNET	Référence	4 711 420	0%
	Entropie locale	5 108 906	0,191%
	Prot. de bavardage	**2 075 012**	**0,133%**

TABLE 5.10 – Résultats des méthodes d'auto-adaptation pour les traces NorduGrid et SHARCNET avec une limitation de capacité de tamisage à 20.

Avec cette limitation de capacité, l'adaptation par entropie locale montre des temps d'exécution légèrement accrus, en raison du phénomène déjà évoqué : une capacité d'adaptation insuffisante après les pics de surcharge. Pour le protocole de bavardage, les résultats sont plus nuancés : la trace SHARCNET montre une légère dégradation, mais sur la trace NorduGrid, le temps d'exécution est inférieur à celui observé sans limitation. Ce phénomène s'explique par une tension plus élevée et plus homogène dans le système, qui favorise une meilleure répartition des grains et une utilisation plus régulière des cellules.

Dans l'ensemble, quelle que soit la méthode, la consommation énergétique associée aux avalanches pour l'équilibrage de la charge reste marginale, y compris en présence de cette contrainte matérielle.

En outre, ces résultats font écho à l'étude de la section précédente : l'adaptation par entropie locale est axée

sur la parcimonie de l'utilisation des ressources, tandis que l'adaptation par protocole de bavardage offre une bien meilleure qualité de service.

5.7 Conclusion

Dans ce chapitre, nous avons introduit et étudié une évolution du modèle du tamis, lui-même dérivé du modèle du tas de sable de Bak, Tang et Wiesenfeld : le tamis auto-adaptatif. Cette évolution vise à doter le tamis de capacités d'adaptation, lui permettant de fonctionner efficacement dans un environnement limité, à l'image des systèmes de traitement réels. Deux mécanismes principaux ont été intégrés au modèle : les seuils critiques dynamiques et les capacités de tamisage adaptatives.

Le premier mécanisme, fondé sur l'ajustement dynamique des seuils critiques d'éboulement, permet au système de stocker temporairement davantage de grains afin de stopper des avalanches qui, autrement, seraient interminables. Cette régulation a également pour effet de réduire drastiquement l'ampleur des avalanches, c'est-à-dire le nombre de cellules impliquées. Ces résultats contribuent non seulement à la faisabilité du modèle dans un environnement limité, mais aussi à la réduction de la consommation énergétique liée aux processus d'avalanches pour redistribuer les grains de sable.

Le second mécanisme concerne l'adaptation dynamique de la capacité de tamisage des cellules, de manière à permettre au système de s'ajuster face à n'importe quel niveau de charge. À cette fin, deux approches ont été proposées : l'adaptation par entropie locale, et l'adaptation par protocole de bavardage.

L'adaptation par entropie locale, fondée sur une mesure locale du mouvement des grains, offre des performances proches de l'optimal dans les scénarios de surcharge constante. En revanche, elle ne surpasse pas les références dans les scénarios à charge réaliste, du fait de la mesure du mouvement sur une fenêtre glissante. Elle présente toutefois une gestion fine et contrôlée des capacités, avantageuse pour des environnements maîtrisés.

L'adaptation par protocole de bavardage affiche également des performances quasi-optimales en surcharge constante, tout en obtenant des temps d'exécution significativement réduits face aux charges réalistes. Cependant, sa grande réactivité réduit la tension locale dans le système, ce qui peut conduire à une moins bonne répartition de la charge.

Dans les deux cas, les méthodes proposées permettent au système de s'ajuster de manière décentralisée, et de réaliser efficacement le tamisage selon la charge soumise. L'adaptation par entropie locale offre une meilleure parcimonie de l'utilisation des ressources, tandis que l'adaptation par protocole de bavardage propose une meilleure qualité de service.

Néanmoins, l'approche par entropie locale présente une limite notable : elle nécessite une calibration préalable des paramètres, soit à partir d'une analyse post-mortem, soit en disposant d'une connaissance anticipée de la charge. Elle offre ainsi une adaptation précise mais présentant une certaine rigidité, bien qu'elle soit capable de gérer des charges aléatoires comme nous l'avons vu. À l'inverse, dans un contexte de charge incertaine et s'il n'est pas possible de conduire une étude post-mortem, l'adaptation par protocole de bavardage constitue une solution plus souple et robuste, capable de maintenir les performances sans connaissance a priori.

Chapitre 6

Conclusion

Les systèmes numériques sont généralement surdimensionnés afin de pouvoir encaisser les pics de charge, ce qui entraîne bien souvent une consommation énergétique excédentaire. Les performances globales d'un tel système dépendent notamment de la qualité de l'équilibrage de la charge, en particulier dans des contextes de ressources limitées. C'est dans cette perspective que nous avons cherché, au cours de ces travaux, à concevoir des mécanismes capables d'optimiser la consommation énergétique tout en préservant la qualité de service, même en conditions de surcharge.

Nous avons d'abord étudié les grands paradigmes de l'équilibrage de charge, en abordant les environnements d'exécution, les architectures concernées ainsi que les modes de prise de décision afin de pouvoir situer notre travail. Nous avons également passé en revue quelques métriques d'évaluation des algorithmes et analysé en détail certains mécanismes spécifiques. Les approches auto-organisatrices se sont révélées particulièrement intéressantes en raison de leur robustesse naturelle, de leur parcimonie, et de leur capacité d'adaptation locale via des interactions entre ressources, les rendant bien adaptées à des environnements décentralisés, dynamiques et de grande échelle.

Inspirés par les mécanismes d'auto-organisation présents dans la nature, nous nous sommes ensuite penchés sur le concept de criticalité auto-organisée, un état dynamique au bord de l'équilibre entre ordre et chaos. Ce principe a été étudié à travers le modèle du tas de sable, et l'une de ses variantes, le tamis, démontrant des capacités prometteuses en termes d'équilibrage distribué.

Notre première contribution porte sur l'étude de la robustesse de la criticalité auto-organisée dans le modèle du tas de sable canonique. Pour cela, nous avons proposé un cadre expérimental original, intégrant : (i) un nouvel algorithme de recâblage, permettant de conserver le fonctionnement du modèle canonique ; (ii) un algorithme de dégradation progressive de la structure sous-jacente, simulant des défaillances matérielles. Ces outils nous ont permis d'analyser le comportement du système sur différentes topologies, allant de grilles régulières à des graphes aléatoires ou de type petit-monde, ainsi que leurs variantes dégradées. Les résultats montrent que les structures faiblement régulières, notamment les petits-mondes, permettent une circulation plus efficace de l'énergie et retardent significativement l'effondrement du système, suggérant pourquoi de telles structures émergent plus naturellement dans les systèmes biologiques ou sociaux.

Plus largement, la robustesse structurelle ouvre la voie à des applications plus concrètes dans plusieurs domaines où les environnements sont partiellement défaillants, dynamiques ou difficiles à contrôler. On peut citer notamment :

— les réseaux de capteurs sans fil, sujets à des pertes fréquentes de nœuds ou de liens ;
— les systèmes pair-à-pair ou blockchains, où la topologie est évolutive et décentralisées ;
— les infrastructures informatiques tactiques ou embarquées, utilisées en environnement dégradé (militaire ou spatial par exemple) ;
— les réseaux neuronaux biologiques ou bio-inspirés, où la plasticité structurelle est naturelle ;
— les systèmes d'équilibrage de charge distribués, dans le cloud ou à la périphérie (edge computing), où des machines peuvent apparaître et disparaître sans coordination centrale.

En résumé, cette étude démontre que la dynamique du tas de sable peut servir de fondement à des mécanismes de régulation distribuée efficaces, même lorsque la structure sous-jacente est instable ou fragmentée, ce qui constitue un atout fort pour le design de systèmes autonomes robustes.

La seconde contribution concerne l'extension du modèle du tamis à des environnements finis, où le nombre de ressources est limité. Ce contexte soulève un enjeu critique : la gestion de la surcharge. Dans le modèle classique, une charge excessive entraîne une saturation du tamis, provoquant une explosion de la consommation énergétique liée au mécanisme d'équilibrage (avalanches) sans réel bénéfice.

Pour répondre à cette limitation, nous avons introduit le tamis auto-adaptatif, intégrant deux mécanismes décentralisés : une adaptation du seuil critique d'éboulement des cellules, et une adaptation des capacités de tamisage (puissance locale des ressources). Ces adaptations sont déclenchées localement, à partir des mouvements de grains observés, permettant une réaction intrinsèque du système aux variations de charge. Le seuil critique dynamique permet d'amortir les avalanches en autorisant temporairement l'accumulation de grains sur les cellules, réduisant ainsi la consommation associée. Pour l'adaptation des capacités, deux stratégies ont été développées : l'une basée sur l'entropie locale (mesure du désordre), l'autre sur un protocole de bavardage.

Les deux méthodes se sont révélées efficaces pour faire face à tous les types de charges (constantes, fluctuantes ou réelles) en maintenant une consommation proche de l'optimum, c'est-à-dire juste suffisante pour absorber la charge. Le protocole de bavardage, sans paramètre, présente l'avantage de la simplicité et de la robustesse. L'approche par entropie locale, quant à elle, permet une adaptation plus fine et contrôlée, au prix d'un réglage de paramètres.

En résumé, ces travaux ouvrent la voie à une gestion énergétique intelligente et décentralisée, fondée sur des principes naturels d'auto-organisation. Ils démontrent que des modèles simples, bien conçus, peuvent offrir une robustesse structurelle, une adaptabilité dynamique, et une efficacité énergétique dans des environnements complexes, sans recours à une supervision centralisée.

Perspectives

Plusieurs pistes peuvent être envisagées pour prolonger et approfondir ces travaux. Nous présentons ici quelques perspectives de recherche.

Seuil critique dynamique Le mécanisme d'adaptation des seuils critiques actuel ne permet une diminution de ces seuils que lorsqu'une avalanche de faible intensité survient. Cela peut conduire à une situation où, si la charge dans le système diminue, les seuils restent durablement élevés en l'absence d'avalanches, celles-ci étant justement inhibées par les seuils élevés. Ce blocage dynamique limite la capacité du système à retrouver un bon niveau d'auto-organisation. Il serait ainsi pertinent d'introduire un mécanisme de décrément temporel, faisant décroître progressivement les seuils s'ils n'ont pas été modifiés depuis un certain temps, rétablissant ainsi la plasticité nécessaire à une adaptation continue.

Capacités de tamisage adaptatives Les mécanismes d'adaptation des capacités de tamisage ont principalement été évalués dans des scénarios de surcharge constante. Il serait intéressant d'étendre ces expérimentations à des charges plus réelles, notamment des charges intermittentes, corrélées ou bruitées, afin d'évaluer la robustesse des modèles dans des contextes plus proches du fonctionnement de systèmes numériques réels. Nous avons également introduit de façon préliminaire une limitation arbitraire des capacités pour simuler les contraintes matérielles d'un système physique. Cette contrainte mériterait d'être approfondie, tant du point de vue de sa modélisation que de son impact sur la dynamique globale du tamis auto-adaptatif.

Consommation énergétique Sur le plan des mesures de performance, notre modélisation de la consommation énergétique reste très simplifiée. Ce choix assumé a permis de se concentrer sur la dynamique du système, mais il s'éloigne du comportement réel des ressources de calcul, dont la consommation n'évolue pas linéairement avec la puissance mobilisée. Une modélisation plus fine de la consommation énergétique, par exemple inspirée de profils de consommation de CPU, permettrait de mieux distinguer les performances des différents mécanismes d'adaptation. Dans ce cadre, il est probable que la stratégie du protocole de bavardage, qui induit des pics plus marqués de capacité, s'avère moins efficaces que l'approche basée sur l'entropie locale, en termes de rendement énergétique global.

Avalanches avec tabou Un mécanisme de déplacement avec tabou a été brièvement exploré au cours de nos travaux, bien qu'il n'ait pas été présenté dans ce document. Ce mécanisme intervient lors de la propagation des avalanches, en orientant la réaffectation des grains vers les cellules voisines les plus stables, c'est-à-dire celles qui ont tendance à donner peu de grains. L'objectif est d'éviter de réinjecter de la charge dans une cellule instable, susceptible de s'ébouler à nouveau, et de privilégier les cellules relativement sous-chargées pour améliorer la stabilité locale. Un facteur aléatoire est intégré dans la sélection des cellules receveuses afin de maintenir une diversité dans les chemins empruntés par les grains, évitant ainsi la formation de canaux rigides dans le système.

Les premiers résultats expérimentaux obtenus suggèrent que ce mécanisme présente plusieurs avantages notables :

— il réduit le nombre total de déplacements de grains sans perturber l'état de criticalité auto-organisée ;
— il produit un équilibrage quasi parfait de la charge entre les cellules ;
— il améliore nettement la robustesse du tamis sans mécanismes d'adaptation, lui permettant de supporter

une charge allant jusqu'à sa taille moins un, sans déclencher d'avalanche infinie, contre environ 97% de sa taille sans déplacement avec tabou.

Ce mécanisme s'annonce donc particulièrement prometteur et mériterait une étude approfondie, tant sur le plan de son intégration aux autres mécanismes adaptatifs que sur celui de son impact global sur la performance du système.

Qualité de service Le modèle du tamis, y compris dans sa version adaptative, ne prend pas en compte l'ancienneté des grains dans les files des cellules. Cela peut entraîner un délai du traitement de grains anciens, en particulier lors des avalanches, et dégrader la qualité de service. Un mécanisme d'ordonnancement pourrait être introduit pour favoriser le traitement des grains les plus anciens.

Dans cette même optique, il serait pertinent de limiter le nombre de déplacements d'un grain afin d'éviter qu'il soit constamment repositionné en fin de file. Cela pourrait se faire via une organisation en deux files, comme l'algorithme de vol de travail étudié au Chapitre 2 : une file contenant les grains devenus immobiles, et une autre pour ceux encore déplaçables. La cellule prioriserait alors la première pour sélectionner les grains à tamiser. Des règles spécifiques pourraient toutefois permettre le redéploiement exceptionnel de grains normalement fixes, notamment en cas de surcharge locale. Un tel mécanisme influencerait nécessairement la dynamique des avalanches, et donc l'état d'auto-organisation du système, ce qui en fait un sujet d'étude à part entière.

Structure sous-jacente Les expérimentations menées ont été limitées à des structures de grille régulière de taille modérée, en raison du coût des simulations. Il serait pertinent d'étudier le comportement du tamis auto-adaptatif dans des structures non régulières (petit monde, aléatoires, sans échelle, etc.) ou de plus grande dimension. Il serait notamment possible de coupler les résultats du Chapitre 4, consacré à la robustesse du tas de sable, avec les mécanismes du tamis auto-adaptatif pour évaluer leur efficacité dans des environnements plus hétérogènes ou dégradés.

Concrétisation du modèle Enfin, bien que le tamis auto-adaptatif soit un modèle abstrait, ses mécanismes pourraient être formalisés dans un cadre applicatif concret. Un exemple pertinent serait celui des services cloud, où la variation de la demande est absorbée par la création ou la suppression de répliques de services. Dans ce contexte, chaque réplique peut être assimilée à une unité de capacité de tamisage, contrainte par les ressources physiques du serveur. Une telle formalisation rapprocherait le modèle des systèmes réels de gestion d'élasticité, et permettrait d'exploiter les bénéfices des principes d'auto-organisation et de criticalité dans la conception de systèmes numériques autonomes et résilients.

Asynchronicité des avalanches L'ensemble des études menées dans ce travail repose sur un cadre de simulation synchrone. Cela signifie que lorsqu'une avalanche se déclenche, la simulation est suspendue et l'avalanche est intégralement traitée jusqu'à ce que le système retrouve un état stable, avant de reprendre son évolution normale. Or, les systèmes réels ne fonctionnent pas de manière aussi séquentielle : le traitement des tâches et l'équilibrage de charge s'y déroulent de manière parallèle et continue. Adopter un paradigme d'avalanches

asynchrones reviendrait à faire évoluer profondément le cadre d'étude, car les avalanches pourraient alors se superposer et interagir, rendant leur mesure (durée, amplitude) et leur analyse bien plus complexes. De plus, le traitement simultané des tâches interférerait directement avec la dynamique des avalanches, modifiant ainsi les mécanismes d'équilibrage de charge.

Le paradigme asynchrone a été envisagé dès les débuts de ce travail. Cependant, en raison de son éloignement conceptuel par rapport au modèle canonique du tas de sable, nous avons choisi de le mettre temporairement de côté. Il constitue néanmoins une piste de recherche essentielle, nécessitant une attention approfondie.

Bibliographie

AFZAL, S., & KAVITHA, G. (2019). Load balancing in cloud computing – a hierarchical taxonomical classification. *Journal of Cloud Computing*, *8*(1), 22. https://doi.org/10.1186/s13677-019-0146-7

ALAKEEL, A. (2009). A Guide to Dynamic Load Balancing in Distributed Computer Systems. *International Journal of Computer Science and Network Security (IJCSNS)*, *10*.

ALENCAR, A. M., ANDRADE, J. S., & LUCENA, L. S. (1997). Self-organized percolation. *Physical Review E*, *56*(3), R2379-R2382. https://doi.org/10.1103/PhysRevE.56.R2379

AL-RAYIS, E., & KURDI, H. (2013). Performance Analysis of Load Balancing Architectures in Cloud Computing. *2013 European Modelling Symposium*, 520-524. https://doi.org/10.1109/EMS.2013.10

ARABNEJAD, H., & BARBOSA, J. G. (2014). List Scheduling Algorithm for Heterogeneous Systems by an Optimistic Cost Table. *IEEE Transactions on Parallel and Distributed Systems*, *25*(3), 682-694. https://doi.org/10.1109/TPDS.2013.57

AZAR, Y., BRODER, A. Z., KARLIN, A. R., & UPFAL, E. (1994). Balanced Allocations. *Proceedings of the twenty-sixth annual ACM symposium on theory of computing*, 593-602.

BAK, P. (1996). *How nature works : the science of self-organized criticality*. Springer New York. https://doi.org/10.1007/978-1-4757-5426-1

BAK, P., TANG, C., & WIESENFELD, K. (1987). Self-organized criticality : an explanation of the $1/f$ noise. *Physical Review Letters*, *59*(4), 381-384. https://doi.org/10.1103/PhysRevLett.59.381

BAK, P., TANG, C., & WIESENFELD, K. (1988). Self-organized criticality. *Physical Review A*, *38*(1), 364-374. https://doi.org/10.1103/PhysRevA.38.364

BARABÁSI, A.-L., & ALBERT, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509-512. https://doi.org/10.1126/science.286.5439.509

BASSETT, D. S., & BULLMORE, E. (2006). Small-world brain networks. *The Neuroscientist*, *12*(6), 512-523. https://doi.org/10.1177/1073858406293182

BEGGS, J. M., & PLENZ, D. (2003). Neuronal avalanches in neocortical circuits. *The Journal of Neuroscience*, *23*(35), 11167-11177. https://doi.org/10.1523/JNEUROSCI.23-35-11167.2003

BELGAUM, M. R., MUSA, S., ALAM, M. M., & SU'UD, M. M. (2020). A Systematic Review of Load Balancing Techniques in Software-Defined Networking. *IEEE Access*, *8*, 98612-98636. https://doi.org/10.1109/ACCESS.2020.2995849

BHAUMIK, H., & SANTRA, S. B. (2013). Critical properties of a dissipative sandpile model on small-world networks. *Physical Review E*, *88*(6). https://doi.org/https://doi.org/10.1103/PhysRevE.88.062817

BIESMEIJER, J. C., & SEELEY, T. D. (2005). The use of waggle dance information by honey bees throughout their foraging careers. *Behavioral Ecology and Sociobiology*, *59*(1), 133-142. https://doi.org/10.1007/s00265-005-0019-6

BIONDO, A. E., PLUCHINO, A., & RAPISARDA, A. (2015). Modeling financial markets by self-organized criticality. *Physical Review E*, *92*(4), 042814. https://doi.org/10.1103/PhysRevE.92.042814

BIRATTARI, M., DI CARO, G., & DORIGO, M. (2002). Toward the formal foundation of ant programming. In M. DORIGO, G. DI CARO & M. SAMPELS (Éd.), *Ant algorithms* (p. 188-201, T. 2463). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-45724-0_16

BIRATTARI, M., & DORIGO, M. (2000). For a Formal Foundation of the Ant Programming Approach to Combinatorial Optimization. Part 1 : The problem, the representation, and the general solution strategy. *Technical Report TR-H-301 of the ATR-Human Information Processing Labs*.

BJÖRNER, A., LOVÁSZ, L., & SHOR, P. W. (1991). Chip-firing games on graphs. *European Journal of Combinatorics*, *12*(4), 283-291. https://doi.org/10.1016/S0195-6698(13)80111-4

BLUM, C. (2005). Ant colony optimization : introduction and recent trends. *Physics of Life Reviews*, *2*(4), 353-373. https://doi.org/10.1016/j.plrev.2005.10.001

BLUMOFE, R. D., & LEISERSON, C. E. (1999). Scheduling multithreaded computations by work stealing. *Journal of the ACM*, *46*(5), 720-748. https://doi.org/10.1145/324133.324234

BONABEAU, E., THERAULAZ, G., DENEUBOURG, J.-L., ARON, S., & CAMAZINE, S. (1997). Self-organization in social insects. *Trends in Ecology & Evolution*, *12*(5), 188-193. https://doi.org/10.1016/S0169-5347(97)01048-3

BOOLCHAND, P., LUCOVSKY, G., PHILLIPS, J. C., & THORPE, M. F. (2005). Self-organization and the physics of glassy networks. *Philosophical Magazine*, *85*(32), 3823-3838. https://doi.org/10.1080/14786430500256425

BORNHOLDT, S., & RÖHL, T. (2003). Self-organized critical neural networks. *Physical Review E*, *67*(6), 066118. https://doi.org/10.1103/PhysRevE.67.066118

BOULMIER, A., ABDENNADHER, N., & CHOPARD, B. (2022). Optimal load balancing and assessment of existing load balancing criteria. *Journal of Parallel and Distributed Computing*, *169*, 211-225. https://doi.org/10.1016/j.jpdc.2022.07.002

BOULMIER, A., RAYNAUD, F., ABDENNADHER, N., & CHOPARD, B. (2019). On the Benefits of Anticipating Load Imbalance for Performance Optimization of Parallel Applications. *2019 IEEE International Conference on Cluster Computing (CLUSTER)*, 1-9. https://doi.org/10.1109/CLUSTER.2019.8890998

BRODAL, G. S., SIOUTAS, S., TSICHLAS, K., & ZAROLIAGIS, C. (2015). D^2-Tree : a new overlay with deterministic bounds. *Algorithmica*, *72*(3), 860-883. https://doi.org/10.1007/s00453-014-9878-4

BURRIDGE, R., & KNOPOFF, L. (1967). Model and theoretical seismicity. *Bulletin of the Seismological Society of America*, *57*(3), 341-371. https://doi.org/10.1785/BSSA0570030341

CAJUEIRO, D. O., & ANDRADE, R. F. S. (2010). Controlling self-organized criticality in sandpile models. *Physical Review E*, *81*(1), 015102. https://doi.org/10.1103/PhysRevE.81.015102

CARDON, A., DUTOT, A., GUINAND, F., & OLIVIER, D. (2006). Competing ants for organization detection application to dynamic distribution. In M. AZIZ-ALAOUI & C. BERTELLE (Éd.), *Emergent properties in natural and artificial dynamical systems* (p. 25-52). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-34824-7_2

CARLSON, J. M., & DOYLE, J. (2002). Complexity and robustness. *Proceedings of the National Academy of Sciences*, *99*, 2538-2545. https://doi.org/10.1073/pnas.012582499

CHEKURI, C., & KHANNA, S. (2005). A polynomial time approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, *35*(3), 713-728. https://doi.org/10.1137/S0097539700382820

CROES, G. A. (1958). A Method for Solving Traveling-Salesman Problems. *Operations Research*, *6*(6), 791-812.

de ARCANGELIS, L., & HERRMANN, H. (2002). Self-organized criticality on small world networks. *Physica A : Statistical Mechanics and its Applications*, *308*(1), 545-549. https://doi.org/10.1016/S0378-4371(02)00549-6

de ARCANGELIS, L., PERRONE-CAPANO, C., & HERRMANN, H. J. (2006). Self-organized criticality model for brain plasticity. *Physical Review Letters*, *96*(2), 028107. https://doi.org/10.1103/PhysRevLett.96.028107

DEEPA, T., & CHEELU, D. (2017). A comparative study of static and dynamic load balancing algorithms in cloud computing. *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, 3375-3378. https://doi.org/10.1109/ICECDS.2017.8390086

DEKKER, A. H., & COLBERT, B. D. (2004). Network robustness and graph topology. *Proceedings of the 27th Australasian Conference on Computer Science*, *26*, 359-368.

DEVI, D. C., & UTHARIARAJ, V. R. (2016). Load balancing in cloud computing environment using improved weighted round robin algorithm for nonpreemptive dependent tasks. *The Scientific World Journal*, *2016*, 1-14. https://doi.org/10.1155/2016/3896065

DORIGO, M., BIRATTARI, M., & STUTZLE, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(4), 28-39. https://doi.org/10.1109/MCI.2006.329691

DORIGO, M., BONABEAU, E., & THERAULAZ, G. (2000). Ant algorithms and stigmergy. *Future Generation Computer Systems*, *16*(8), 851-871. https://doi.org/10.1016/S0167-739X(00)00042-X

DROSSEL, B., & SCHWABL, F. (1992). Self-organized critical forest-fire model. *Physical Review Letters*, *69*(11), 1629-1632. https://doi.org/10.1103/PhysRevLett.69.1629

DUTOT, A. (2005). *Distribution dynamique adaptative à l'aide de mécanismes d'intelligence collective* [Thèse]. Université du Havre.

ERDÖS, P., & RÉNYI, A. (1960). On the evolution of random graphs. *Publ. math. inst. hung. acad. sci*, *5*(1), 17-60.

EYTAN, D., & MAROM, S. (2006). Dynamics and effective topology underlying synchronization in networks of cortical neurons. *The Journal of Neuroscience*, *26*(33), 8465-8476. https://doi.org/10.1523/JNEUROSCI.1627-06.2006

GĄSIOR, J., & SEREDYŃSKI, F. (2017). A sandpile cellular automata-based scheduler and load balancer. *Journal of Computational Science*, *21*, 460-468. https://doi.org/10.1016/j.jocs.2016.08.005

GILMAN, D. L., FUGLISTER, F. J., & MITCHELL, J. M. (1963). On the Power Spectrum of "Red Noise". *Journal of Atmospheric Sciences*, *20*(2), 182-184. https://doi.org/10.1175/1520-0469(1963)020<0182:OTPSON>2.0.CO;2

GOH, K.-I., KAHNG, B., & KIM, D. (2001). Universal behavior of load distribution in scale-free networks. *Physical Review Letters*, *87*(27), 278701. https://doi.org/10.1103/PhysRevLett.87.278701

GOH, K.-I., LEE, D.-S., KAHNG, B., & KIM, D. (2003). Sandpile on scale-free networks. *Physical Review Letters*, *91*(14), 148701. https://doi.org/10.1103/PhysRevLett.91.148701

GOLDSZTAJN, D., BORST, S. C., VAN LEEUWAARDEN, J. S. H., MUKHERJEE, D., & WHITING, P. A. (2022). Self-learning threshold-based load balancing. *INFORMS Journal on Computing*, *34*(1), 39-54. https://doi.org/10.1287/ijoc.2021.1100

GONZALEZ, R., & HOROWITZ, M. (1996). Energy dissipation in general purpose microprocessors. *IEEE Journal of Solid-State Circuits*, *31*(9), 1277-1284. https://doi.org/10.1109/4.535411

GRIBBLE, S. (2001). Robustness in complex systems. *Proceedings Eighth Workshop on Hot Topics in Operating Systems*, 21-26. https://doi.org/10.1109/HOTOS.2001.990056

GROSS, T., & BLASIUS, B. (2008). Adaptive coevolutionary networks : a review. *Journal of The Royal Society Interface*, *5*(20), 259-271. https://doi.org/10.1098/rsif.2007.1229

GURES, E., SHAYEA, I., ERGEN, M., AZMI, M. H., & EL-SALEH, A. A. (2022). Machine Learning-Based Load Balancing Algorithms in Future Heterogeneous Networks : A Survey. *IEEE Access*, *10*, 37689-37717. https://doi.org/10.1109/ACCESS.2022.3161511

HAHN, G., PETERMANN, T., HAVENITH, M. N., YU, S., SINGER, W., PLENZ, D., & NIKOLIĆ, D. (2010). Neuronal avalanches in spontaneous activity in vivo. *Journal of Neurophysiology*, *104*(6), 3312-3322. https://doi.org/10.1152/jn.00953.2009

HASAN, M. S., ALVARES, F., LEDOUX, T., & PAZAT, J.-L. (2017). Investigating Energy Consumption and Performance Trade-Off for Interactive Cloud Application. *IEEE Transactions on Sustainable Computing*, *2*(2), 113-126. https://doi.org/10.1109/TSUSC.2017.2714959

HE, K., ROZNER, E., AGARWAL, K., FELTER, W., CARTER, J., & AKELLA, A. (2015). Presto : edge-based load balancing for fast datacenter networks. *ACM SIGCOMM Computer Communication Review*, *45*(4), 465-478. https://doi.org/10.1145/2829988.2787507

HESSE, J., & GROSS, T. (2014). Self-organized criticality as a fundamental property of neural systems. *Frontiers in Systems Neuroscience*, *8*. https://doi.org/10.3389/fnsys.2014.00166

HEYLIGHEN, F. (2009a). Complexity and self-organization. In M. J. BATES & M. N. MAACK (Éd.), *Encyclopedia of library and information sciences, third edition* (3rd Edition). CRC Press. https://doi.org/10.1081/E-ELIS3

HEYLIGHEN, F. (2009b). The Science of Self-Organization and Adaptivity [OCLC : 488796066]. In L. D. KIEL (Éd.), *Knowledge Management, Organizational Intelligence and Learning, and Complexity.* Eolss Publishers.

HIDAYAT, T., AZZERY, Y., & MAHARDIKO, R. (2020). Load Balancing Network by using Round Robin Algorithm : A Systematic Literature Review. *Jurnal Online Informatika*, *4*(2), 85-89. https://doi.org/10.15575/join.v4i2.446

HOCHREITER, S. (1997). Long Short-term Memory. *Neural Computation MIT-Press*.

HU, F., CHEN, L., & CHEN, J. (2021). Robustness evaluation of complex power grids containing renewable energy. *International Journal of Electrical Power & Energy Systems*, *132*, 107187. https://doi.org/10.1016/j.ijepes.2021.107187

IOSUP, A., LI, H., JAN, M., ANOEP, S., DUMITRESCU, C., WOLTERS, L., & EPEMA, D. H. (2008a). The grid workloads archive. *Future Generation Computer Systems*, *24*(7), 672-686. https://doi.org/10.1016/j.future.2008.02.003

IOSUP, A., SONMEZ, O., ANOEP, S., & EPEMA, D. (2008b). The performance of bags-of-tasks in large-scale distributed systems. *Proceedings of the 17th International Symposium on High Performance Distributed Computing*, 97-108. https://doi.org/10.1145/1383422.1383435

ISAEVA, V. V. (2012). Self-organization in biological systems. *Biology Bulletin*, *39*(2), 110-118. https://doi.org/10.1134/S1062359012020069

IVANISENKO, I. N., & RADIVILOVA, T. A. (2015). Survey of major load balancing algorithms in distributed system. *2015 Information Technologies in Innovation Business Conference (ITIB)*, 89-92. https://doi.org/10.1109/ITIB.2015.7355061

JADER, O. H., ZEEBAREE, S., & ZEBARI, R. R. (2019). A state of art survey for web server performance measurement and load balancing mechanisms. *International Journal of Scientific & Technology Research*, *8*(12), 535-543.

JAFARNEJAD GHOMI, E., MASOUD RAHMANI, A., & NASIH QADER, N. (2017). Load-balancing algorithms in cloud computing : a survey. *Journal of Network and Computer Applications*, *88*, 50-71. https://doi.org/10.1016/j.jnca.2017.04.007

JAIN, R. K., CHIU, D.-M. W., HAWE, W. R., et al. (1984). A quantitative measure of fairness and discrimination. *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, *21*(1).

JELASITY, M., MONTRESOR, A., & BABAOGLU, O. (2004). A Modular Paradigm for Building Self-Organizing Peer-to-Peer Applications. In G. DI MARZO SERUGENDO, A. KARAGEORGOS, O. F. RANA & F. ZAMBONELLI (Éd.). G. GOOS, J. HARTMANIS & J. VAN LEEUWEN (Éd.), *Engineering Self-Organising Systems* (p. 265-282, T. 2977). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-24701-2_18

JIE HU, R. K. (2006). Decentralized Load Balancing on Unstructured Peer-2-Peer Computing Grids. *Fifth IEEE International Symposium on Network Computing and Applications (NCA'06)*, 247-250. https://doi.org/10.1109/NCA.2006.21

KANELLOPOULOS, D., & SHARMA, V. (2022). Dynamic load balancing techniques in the IoT : a review. *Symmetry*, *14*(12), 2554. https://doi.org/10.3390/sym14122554

KARMAKAR, R., & MANNA, S. S. (2005). Sandpile model on an optimized scale-free network on Euclidean space. *Journal of Physics A : Mathematical and General*, *38*(6). https://doi.org/10.1088/0305-4470/38/6/L03

KATAL, A., DAHIYA, S., & CHOUDHURY, T. (2023). Energy efficiency in cloud computing data centers : a survey on software technologies. *Cluster Computing*, *26*(3), 1845-1875. https://doi.org/10.1007/s10586-022-03713-0

KATEVENIS, M., SIDIROPOULOS, S., & COURCOUBETIS, C. (1991). Weighted round-robin cell multiplexing in a general-purpose ATM switch chip. *IEEE Journal on Selected Areas in Communications*, *9*(8), 1265-1279. https://doi.org/10.1109/49.105173

KATYAL, M., & MISHRA, A. (2013). A Comparative Study of Load Balancing Algorithms in Cloud Computing Environment. *International Journal of Distributed and Cloud Computing*, *1*(2). https://doi.org/10.48550/ARXIV.1403.6918

KAUR, A., KAUR, B., SINGH, P., DEVGAN, M. S., & TOOR, H. K. (2020). Load Balancing Optimization Based on Deep Learning Approach in Cloud Environment. *International Journal of Information Technology and Computer Science*, *12*(3), 8-18. https://doi.org/10.5815/ijitcs.2020.03.02

KENNEDY, J., & EBERHART, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*, *4*, 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

KITANO, H. (2004). Biological robustness. *Nature Reviews Genetics*, *5*(11), 826-837. https://doi.org/10.1038/nrg1471

KRASICH, M. (2009). How to estimate and use MTTF/MTBF would the real MTBF please stand up ? *2009 Annual Reliability and Maintainability Symposium*, 353-359. https://doi.org/10.1109/RAMS.2009.4914702

KUMAR, P., & KUMAR, R. (2019). Issues and challenges of load balancing techniques in cloud computing : a survey. *ACM Computing Surveys*, *51*(6), 1-35. https://doi.org/10.1145/3281010

LAREDO, J. J., BOUVRY, P., GUINAND, F., DORRONSORO, B., & FERNANDES, C. (2014). The sandpile scheduler : how self-organized criticality may lead to dynamic load-balancing. *Cluster Computing*, *17*(2), 191-204. https://doi.org/10.1007/s10586-013-0328-x

LAREDO, J. J., DORRONSORO, B., PECERO, J., BOUVRY, P., DURILLO, J. J., & FERNANDES, C. (2012). Designing a Self-Organized Approach for Scheduling Bag-of-Tasks. *2012 Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, 315-320. https://doi.org/10.1109/3PGCIC.2012.28

LAREDO, J. J., GUINAND, F., OLIVIER, D., & BOUVRY, P. (2017). Load Balancing at the Edge of Chaos : How Self-Organized Criticality Can Lead to Energy-Efficient Computing. *IEEE Transactions on Parallel and Distributed Systems*, *28*(2), 517-529. https://doi.org/10.1109/TPDS.2016.2582160

Largest Contentful Paint | Lighthouse [Chrome for Developers]. (2020). Accédé le 19 juin 2025, à partir de https://developer.chrome.com/docs/lighthouse/performance/lighthouse-largest-contentful-paint

LEE, Y. C., & ZOMAYA, A. Y. (2012). Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing*, *60*(2), 268-280. https://doi.org/10.1007/s11227-010-0421-3

LEVIN, S. A. (2005). Self-organization and the emergence of complexity in ecological systems. *BioScience*, *55*(12), 1075. https://doi.org/10.1641/0006-3568(2005)055[1075:SATEOC]2.0.CO;2

LI, K., XU, G., ZHAO, G., DONG, Y., & WANG, D. (2011). Cloud Task Scheduling Based on Load Balancing Ant Colony Optimization. *2011 Sixth Annual Chinagrid Conference*, 3-9. https://doi.org/10.1109/ChinaGrid.2011.17

LIN, W., WANG, H., ZHANG, Y., QI, D., WANG, J. Z., & CHANG, V. (2018). A cloud server energy consumption measurement system for heterogeneous cloud environments. *Information Sciences*, *468*, 47-62. https://doi.org/10.1016/j.ins.2018.08.032

LISE, S., & PACZUSKI, M. (2002). Nonconservative earthquake model of self-organized criticality on a random graph. *Physical Review Letters*, *88*(22), 228301. https://doi.org/10.1103/PhysRevLett.88.228301

LIU, L., YANG, Y., LI, L., & SHI, W. (2006). Using Ant Colony Optimization for SuperScheduling in Computational Grid. *2006 IEEE Asia-Pacific Conference on Services Computing (APSCC'06)*, 539-545. https://doi.org/10.1109/APSCC.2006.112

MALCAI, O., SHILO, Y., & BIHAM, O. (2006). Dissipative sandpile models with universal exponents. *Physical Review E*, *73*(5), 056125. https://doi.org/10.1103/PhysRevE.73.056125

MALONE, C., & BELADY, C. (2006). Metrics to Characterize Data Center & IT Equipment Energy Use, Proceedings of Digital Power Forum, Richardson, TX. *Proceedings of the Digital Power Forum*.

MARKOVIĆ, D., & GROS, C. (2014). Power laws and self-organized criticality in theory and nature. *Physics Reports*, *536*(2), 41-74. https://doi.org/10.1016/j.physrep.2013.11.002

MEGHARAJ, G. C., & MOHAN, K. (2013). Two level hierarchical model of load balancing in cloud. *International Journal of Emerging Technology and Advanced Engineering*, *3*(10), 307-311.

MEISEL, C., STORCH, A., HALLMEYER-ELGNER, S., BULLMORE, E., & GROSS, T. (2012). Failure of adaptive self-organized criticality during epileptic seizure attacks (T. BEHRENS, Éd.). *PLoS Computational Biology*, *8*(1), e1002312. https://doi.org/10.1371/journal.pcbi.1002312

MERINO, C. (2005). The chip-firing game. *Discrete Mathematics*, *302*(1), 188-210. https://doi.org/10.1016/j.disc.2004.07.033

MIRJALILI, S. (2019). Genetic Algorithm. In *Evolutionary Algorithms and Neural Networks* (p. 43-55, T. 780). Springer International Publishing. https://doi.org/10.1007/978-3-319-93025-1_4

MISHRA, S. K., SAHOO, B., & PARIDA, P. P. (2020). Load balancing in cloud computing : a big picture. *Journal of King Saud University - Computer and Information Sciences*, *32*(2), 149-158. https://doi.org/10.1016/j.jksuci.2018.01.003

MUHAMMAD RIDZUAN, M. I., & DJOKIC, S. Z. (2019). Energy regulator supply restoration time. *Energies*, *12*(6), 1051. https://doi.org/10.3390/en12061051

MUSTAFA, M. E. (2017). Load Balancing Algorithms Round-Robin (RR), Least-Connection, and Least Loaded Efficiency. *Computer Science & Telecommunications*, *51*(1), 25-30.

MUTHUSAMY, A., & DHANARAJ, R. K. (2023). Dynamic q-learning-based optimized load balancing technique in cloud (D. XU, Éd.). *Mobile Information Systems*, *2023*, 1-16. https://doi.org/10.1155/2023/7250267

NEWMAN, M. J. (2005). A measure of betweenness centrality based on random walks. *Social Networks*, *27*(1), 39-54. https://doi.org/10.1016/j.socnet.2004.11.009

NIELSEN, J. (1993). *Usability engineering*. Academic Press.

OLAMI, Z., FEDER, H. J. S., & CHRISTENSEN, K. (1992). Self-organized criticality in a continuous, non-conservative cellular automaton modeling earthquakes. *Physical Review Letters*, *68*(8), 1244-1247. https://doi.org/10.1103/PhysRevLett.68.1244

PAI, V. S., ARON, M., BANGA, G., SVENDSEN, M., DRUSCHEL, P., ZWAENEPOEL, W., & NAHUM, E. (1998). Locality-Aware Request Distribution in Cluster-Based Network Servers. *Proceedings of the eighth international conference on Architectural support for programming languages and operating systems*, 205-216. https://doi.org/10.1145/291069.291048

PAN, G.-J., ZHANG, D.-M., YIN, Y.-P., & HE, M.-H. (2007). Sandpile on directed small-world networks. *Physica A : Statistical Mechanics and its Applications*, *383*(2), 435-442. https://doi.org/https://doi.org/10.1016/j.physa.2007.04.113

PATEL, D. K., TRIPATHY, D., & TRIPATHY, C. (2016). Survey of load balancing techniques for grid. *Journal of Network and Computer Applications*, *65*, 103-119. https://doi.org/10.1016/j.jnca.2016.02.012

PERC, M. (2013). Self-organization of progress across the century of physics. *Scientific Reports*, *3*(1), 1720. https://doi.org/10.1038/srep01720

PHAM, D., GHANBARZADEH, A., KOC, E., OTRI, S., RAHIM, S., & ZAIDI, M. (2005). The bees algorithm. *Technical Note, Manufacturing Engineering Centre, Cardiff University, UK*, 44-48.

PINAR, A., & AYKANAT, C. (2004). Fast optimal load balancing algorithms for 1d partitioning. *Journal of Parallel and Distributed Computing*, *64*(8), 974-996. https://doi.org/10.1016/j.jpdc.2004.05.003

PLENZ, D., RIBEIRO, T. L., MILLER, S. R., KELLS, P. A., VAKILI, A., & CAPEK, E. L. (2021). Self-Organized Criticality in the Brain. *Frontiers in Physics*, *9*, 639389. https://doi.org/10.3389/fphy.2021.639389

POURGHEBLEH, B., & HAYYOLALAM, V. (2020). A comprehensive and systematic review of the load balancing mechanisms in the internet of things. *Cluster Computing*, *23*(2), 641-661. https://doi.org/10.1007/s10586-019-02950-0

PRIGOGINE, I. (1978). Time, Structure, and Fluctuations. *Science*, *201*(4358), 777-785. https://doi.org/10.1126/science.201.4358.777

PRIYA, B., & GNANASEKARAN, T. (2017). Hierarchical Load Balancing Algorithms in Cloud : A Survey. *International Journal of Computer Applications*, *171*(4), 19-22. https://doi.org/10.5120/ijca2017915018

QI, J., & PFENNINGER, S. (2015). Controlling the self-organizing dynamics in a sandpile model on complex networks by failure tolerance. *Europhysics Letters*, *111*(3). https://doi.org/10.1209/0295-5075/111/38006

QIN, X., LI, B., & YING, L. (2021). Distributed Threshold-based Offloading for Large-Scale Mobile Cloud Computing. *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 1-10. https://doi.org/10.1109/INFOCOM42981.2021.9488821

QIN, X., LI, B., & YING, L. (2023a). Efficient Distributed Threshold-Based Offloading for Large-Scale Mobile Cloud Computing. *IEEE/ACM Transactions on Networking*, *31*(1), 308-321. https://doi.org/10.1109/TNET.2022.3193073

QIN, X., XIE, Q., & LI, B. (2023b). Distributed Threshold-Based Offloading for Heterogeneous Mobile Edge Computing. *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*, 202-213. https://doi.org/10.1109/ICDCS57875.2023.00024

RADHIKA, D., & DURAIPANDIAN, M. (2021). Load Balancing in Cloud Computing Using Support Vector Machine and Optimized Dynamic Task Scheduling. *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 1-6. https://doi.org/10.1109/ICRITO51393.2021.9596289

RADOJEVIĆ, B., & ŽAGAR, M. (2011). Analysis of issues with load balancing algorithms in hosted (cloud) environments. *2011 Proceedings of the 34th international convention MIPRO*, 416-420.

RAHMEH, O. A., JOHNSON, P., & TALEB-BENDIAB, A. (2008). A Dynamic Biased Random Sampling Scheme for Scalable and Reliable Grid Networks. *INFOCOMP Journal of Computer Science*, *7*(4), 1-10.

RAMEZANI, F., LU, J., & HUSSAIN, F. K. (2014). Task-based system load balancing in cloud computing using particle swarm optimization. *International Journal of Parallel Programming*, *42*(5), 739-754. https://doi.org/10.1007/s10766-013-0275-4

RANDLES, M., LAMB, D., & TALEB-BENDIAB, A. (2010). A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing. *2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops*, 551-556. https://doi.org/10.1109/WAINA.2010.85

REYNOLDS, C. W. (1987). Flocks, herds and schools : a distributed behavioral model. *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 25-34. https://doi.org/10.1145/37401.37406

ROHLF, T., & BORNHOLDT, S. (2009). Self-organized criticality and adaptation in discrete dynamical networks [Series Title : Understanding Complex Systems]. In T. GROSS & H. SAYAMA (Éd.), *Adaptive networks* (p. 73-106). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-01284-6_5

ROSS, K. W., & YAO, D. D. (1991). Optimal load balancing and scheduling in a distributed computer system. *Journal of the ACM (JACM)*, *38*(3), 676-689.

ROY, S., HOSSAIN, N., & AL ASIF, M. R. (2019). Measuring the performance on load balancing algorithms. *Global Journal of Computer Science and Technology : B Cloud and Distributed*, *19*(1).

RYBARSCH, M., & BORNHOLDT, S. (2014). Avalanches in self-organized critical neural networks : a minimal model for the neural SOC universality class (D. R. CHIALVO, Éd.). *PLoS ONE*, *9*(4), e93090. https://doi.org/10.1371/journal.pone.0093090

SAFFRE, F., TATESON, R., HALLOY, J., SHACKLETON, M., & DENEUBOURG, J. L. (2009). Aggregation dynamics in overlay networks and their implications for self-organized distributed applications. *The Computer Journal*, *52*(4), 397-412. https://doi.org/10.1093/comjnl/bxn017

SESUM-CAVIC, V., & KÜHN, E. (2010a). Applying Swarm Intelligence Algorithms for Dynamic Load Balancing to a Cloud Based Call Center. *2010 Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, 255-256. https://doi.org/10.1109/SASO.2010.19

SESUM-CAVIC, V., & KÜHN, E. (2010b). Comparing Configurable Parameters of Swarm Intelligence Algorithms for Dynamic Load Balancing. *2010 Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshop*, 42-49. https://doi.org/10.1109/SASOW.2010.12

SHAFIQ, D. A., JHANJHI, N., & ABDULLAH, A. (2022). Load balancing techniques in cloud computing environment : a review. *Journal of King Saud University - Computer and Information Sciences*, *34*(7), 3910-3933. https://doi.org/10.1016/j.jksuci.2021.02.007

SHENGWEI MEI, YIXIN NI, GANG WANG & SHENGYU WU. (2008). A Study of Self-Organized Criticality of Power System Under Cascading Failures Based on AC-OPF With Voltage Stability Margin. *IEEE Transactions on Power Systems*, *23*(4), 1719-1726. https://doi.org/10.1109/TPWRS.2008.2002295

SIOUTAS, S., SOURLA, E., TSICHLAS, K., VONITSANOS, G., & ZAROLIAGIS, C. (2022). A dynamic distributed deterministic load-balancer for decentralized hierarchical infrastructures. *Algorithms*, *15*(3), 96. https://doi.org/10.3390/a15030096

SORNETTE, A., & SORNETTE, D. (1989). Self-Organized Criticality and Earthquakes. *Europhysics Letters (EPL)*, *9*(3), 197-202. https://doi.org/10.1209/0295-5075/9/3/002

STEIN, S., & KLOSKO, E. (2002). 7 earthquake mechanisms and plate tectonics. In *International geophysics* (p. 69-78, T. 81). Elsevier. https://doi.org/10.1016/S0074-6142(02)80210-8

STROGATZ, S. (2024). *Nonlinear dynamics and chaos : with applications to physics, biology, chemistry, and engineering* (Third edition). CRC Press, Taylor & Francis Group.

TABATABAEE, S. M., LE BOUDEC, J.-Y., & BOYER, M. (2021). Interleaved weighted round-robin : a network calculus analysis. *IEICE Transactions on Communications*, *E104.B*(12), 1479-1493. https://doi.org/10.1587/transcom.2021ITI0001

TANTAWI, A. N., & TOWSLEY, D. (1985). Optimal static load balancing in distributed computer systems. *Journal of the ACM*, *32*(2), 445-465. https://doi.org/10.1145/3149.3156

THE KUBERNETES AUTHORS. (2025). *Autoscaling workloads* [Kubernetes]. Accédé le 29 mai 2025, à partir de https://kubernetes.io/docs/concepts/workloads/autoscaling/

THERAULAZ, G., & BONABEAU, E. (1999). A brief history of stigmergy. *Artificial Life*, *5*(2), 97-116. https://doi.org/10.1162/106454699568700

TURCOTTE, D. L. (1999). Self-organized criticality. *Reports on Progress in Physics*, *62*(10), 1377-1429. https://doi.org/10.1088/0034-4885/62/10/201

UCHECHUKWU, A., LI, K., SHEN, Y., et al. (2014). Energy consumption in cloud computing data centers. *International Journal of Cloud Computing and Services Science (IJ-CLOSER)*, *3*(3), 31-48.

Un fonctionnement écoresponsable - Centres de données Google [Google Data Centers]. (s. d.). Accédé le 19 juin 2025, à partir de https://datacenters.google/intl/fr_ALL/operating-sustainably/

VAN LAARHOVEN, P. J. M., & AARTS, E. H. L. (1987). Simulated annealing. In *Simulated annealing : theory and applications* (p. 7-15). Springer Netherlands. https://doi.org/10.1007/978-94-015-7744-1_2

VÁZQUEZ, A., & COSTA, O. S. (1999). Self-organized criticality and directed percolation. *Journal of Physics A : Mathematical and General*, *32*(14), 2633-2644. https://doi.org/10.1088/0305-4470/32/14/004

VOSS, R. F., & CLARKE, J. (1975). '1/f noise' in music and speech. *Nature*, *258*(5533), 317-318. https://doi.org/10.1038/258317a0

WANG, K., ZHOU, X., LI, T., ZHAO, D., LANG, M., & RAICU, I. (2014). Optimizing load balancing and data-locality with data-aware scheduling. *2014 IEEE International Conference on Big Data (Big Data)*, 119-128. https://doi.org/10.1109/BigData.2014.7004220

WANG, S.-J., & ZHOU, C. (2012). Hierarchical modular structure enhances the robustness of self-organized criticality in neural networks. *New Journal of Physics*, *14*(2), 023005. https://doi.org/10.1088/1367-2630/14/2/023005

WANG, S.-C., YAN, K.-Q., LIAO, W.-P., & WANG, S.-S. (2010). Towards a Load Balancing in a three-level cloud computing network. *2010 3rd International Conference on Computer Science and Information Technology*, *1*, 108-113. https://doi.org/10.1109/ICCSIT.2010.5563889

WATKINS, C. J. C. H., & DAYAN, P. (1992). Q-learning. *Machine Learning*, *8*(3), 279-292. https://doi.org/10.1007/BF00992698

WATTS, D. J., & STROGATZ, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, *393*(6684), 440-442. https://doi.org/10.1038/30918

WILLEBEEK-LEMAIR, M., & REEVES, A. (1993). Strategies for dynamic load balancing on highly parallel computers. *IEEE Transactions on Parallel and Distributed Systems*, *4*(9), 979-993. https://doi.org/10.1109/71.243526

WONG, L.-P., LOW, M. Y. H., & CHONG, C. S. (2008). A Bee Colony Optimization Algorithm for Traveling Salesman Problem. *2008 Second Asia International Conference on Modelling & Simulation (AMS)*, 818-823. https://doi.org/10.1109/AMS.2008.27

YADAV, M. P., PAL, N., & YADAV, D. K. (2021). Workload Prediction over Cloud Server using Time Series Data. *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 267-272. https://doi.org/10.1109/Confluence51648.2021.9377032

YUCE, B., PACKIANATHER, M., MASTROCINQUE, E., PHAM, D., & LAMBIASE, A. (2013). Honey bees inspired optimization method : the bees algorithm. *Insects*, *4*(4), 646-662. https://doi.org/10.3390/insects4040646

ZAKI, M., LI, W., & PARTHASARATHY, S. (1996). Customized dynamic load balancing for a network of workstations. *Proceedings of 5th IEEE International Symposium on High Performance Distributed Computing*, 282-291. https://doi.org/10.1109/HPDC.1996.546198

ZHANG, H., FATA, E., & SUNDARAM, S. (2015). A Notion of Robustness in Complex Networks. *IEEE Transactions on Control of Network Systems*, *2*(3), 310-320. https://doi.org/10.1109/TCNS.2015.2413551

ZHANG, H., ZHANG, J., BAI, W., CHEN, K., & CHOWDHURY, M. (2017). Resilient datacenter load balancing in the wild. *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 253-266. https://doi.org/10.1145/3098822.3098841

ZHENG, G., BHATELÉ, A., MENESES, E., & KALÉ, L. V. (2011). Periodic hierarchical load balancing for large supercomputers. *The International Journal of High Performance Computing Applications*, *25*(4), 371-385. https://doi.org/10.1177/1094342010394383

Annexe A

Résumé des expériences

Cette annexe propose une synthèse des expériences menées dans le cadre des travaux présentés dans le document. Les Tableaux A.1 et A.2 rassemblent respectivement la synthèse des expériences du Chapitre 4 sur la robustesse du tas de sable et du Chapitre 5 sur le tamis auto-adaptatif. Les expériences sont présentées dans l'ordre d'apparition dans les chapitres.

Expérience	**Objectif**	**Méthodologie**	**Résultats**	**Référence**
Recâblage du tas de sable	Étudier l'impact de l'aléatoire dans la structure sur la dynamique du modèle	Recâblage aléatoire progressif de la structure sous-jacente	L'aléatoire améliore le fonctionnement du modèle	Section 4.1.1
Recâblage et dégradation du tas de sable	Étudier l'impact de la dégradation et du recâblage de la structure sur la robustesse du modèle	Recâblage et dégradation progressive de la structure sous-jacente	Le recâblage permet de repousser d'environs 20% de "pannes" le seuil d'effondrement	Section 4.4

TABLE A.1 – Synthèse des expériences sur la robustesse du tas de sable (Chapitre 4).

Expérience	Objectif	Méthodologie	Résultats	Référence
Cas d'étude du seuil critique dynamique	Étudier l'impact du seuil critique dynamique sur les avalanches du tas de sable	Simulation d'avalanches avec différentes configurations initiales du système	Réduction de l'amplitude et de la durée des avalanches ; avalanches "infinies" gérées	Section 5.2.2
Paramétrisation de l'entropie locale	Déterminer le meilleur couple de paramètres pour les deux méthodes	Simulations de 144 couples pour des tailles de grain fixes et fluctuantes	- Adaptation naïve : {0,25 ; 11} - Adaptation prop. : {0,25 ; 14}	Section 5.4.4
Comparaison des méthodes par entropie locale	Étudier les performances des deux méthodes proposées	Simulations de tailles de grain fixes et fluctuantes ; comparaison numérique	Meilleurs contrôle et performances par l'approche proportionnelle	Section 5.4.5
Étude du protocole de bavardage	Étudier les performances du protocole de bavardage pour l'adaptation	Simulations de tailles de grain fixes et fluctuantes ; analyse numérique	Adaptation fonctionnelle et réactive ; peu de mouvements	Section 5.5.3
Comparaison des stratégies d'adaptation	Comparer les approches d'adaptation pour déterminer la meilleure	Analyse numérique du comportement pour des scénarios de charge fixe, fluctuante et réaliste	L'entropie locale est plus parcimonieuse dans l'utilisation des ressources ; Le protocole de bavardage offre une meilleure qualité de service	Section 5.6

TABLE A.2 – Synthèse des expériences sur le tamis auto-adaptatif (Chapitre 5).

Annexe B

Reproductibilité des expériences

Dans cette annexe, nous proposons la marche à suivre pour pouvoir reproduire nos expériences à l'aide du code produit pour nos travaux. Nous verrons dans un premier temps où récupérer le code et les données utilisées. Ensuite, nous présenterons en détails les données issues de systèmes réels utilisés pour les expériences de la Section 5.6. Puis, Enfin, nous verrons comment utiliser le code mis à disposition pour reproduire les expériences.

B.1 Accessibilité du code et des données

Le code est entièrement disponible en open source sous licence MIT. Il est trouvable sur le dépôt Git suivant : https://git.litislab.fr/pheleine/self-adaptive-sand-sieve. Le dépôt comprend tout le code Java du projet, ainsi que les archives contenant les traces d'exécution des systèmes réels. Le code est entièrement documenté afin de faciliter son utilisation. En outre, des scripts Bash sont également inclus, permettant de lancer des expériences en série (batchs) plutôt qu'une à la fois via les classes Java exécutables.

B.2 Les traces d'exécution de systèmes réels

Dans cette section, nous nous intéressons aux données de systèmes réels que nous avons utilisé pour les expériences du Chapitre 5 sur le tamis auto-adaptatif, afin de comparer les deux approches d'adaptation proposées dans nos travaux (Section 5.6).

B.2.1 Informations générales

La charge réelle utilisée dans les expériences est extraite du projet Grid Workload Archive (IOSUP et al., 2008a), dont les contributeurs ont mis à disposition des traces de charge de travail réelles anonymisées issues de plusieurs centres de calcul, dans un objectif de recherche et de validation expérimentale. Les données sont accessibles librement à cette adresse : https://atlarge-research.com/gwa.html (dernière consultation en juillet 2025).

Plus précisément, nous utilisons dans nos expériences les traces provenant des systèmes suivants :

— **AuverGrid :** infrastructure située en Auvergne, composée de 5 clusters totalisant 475 processeurs ;

— **NorduGrid :** système distribué dédié à la recherche académique dans les pays nordiques, comprenant 75 clusters et environ 2000 processeurs ;

— **SHARCNET :** réseau de calcul de haute performance localisé en Ontario (Canada), constitué de 10 clusters pour un total de 6828 processeurs.

Les traces fournies couvrent différentes périodes : l'année 2006 pour AuverGrid, de 2003 à 2006 pour NorduGrid, et de 2005 à 2007 pour SHARCNET. Parmi les nombreuses informations disponibles, deux champs nous intéressent particulièrement : la date de soumission et le temps d'exécution de chaque tâche. Les tâches dont le temps d'exécution est nul (annulées avant traitement) sont écartées de nos expériences.

Toutes les valeurs temporelles sont exprimées en secondes, ce qui permet un alignement direct avec les cycles de simulation de nos expériences : une seconde de la trace équivaut à un cycle de simulation du tamis auto-adaptatif. Chaque tâche est ainsi assimilée à un grain, dont la date de soumission correspond au cycle d'injection dans le tamis, et dont le temps d'exécution représente la taille. La date de soumission étant exprimée en nombre de secondes écoulées depuis le 1er janvier 1970, il est possible de filtrer les tâches par période et de fixer arbitrairement le cycle 0.

B.2.2 Accessibilité des données et leur utilisation

Dans un souci de conservation et de disponibilité des données, les fichiers utilisés au cours de ce travail sont fournis dans le dépôt, en complément du code source. Ils sont regroupés dans une archive compressée nommée `datasets.zip`, située à la racine du dépôt. Par ailleurs, des fichiers d'analyse des traces (également disponibles sur le site https://www.atlarge-research.com/gwa.html) sont inclus afin de fournir des informations complémentaires sur les données utilisées.

Pour pouvoir exploiter ces données dans les simulations, il convient de décompresser l'archive, puis de placer son contenu dans le répertoire de ressources du projet Java, à l'emplacement suivant :

```
src/main/resources/datasets/
```

Une fois en place, les fichiers seront automatiquement chargés par les programmes utilisant la politique de taille de grain fondée sur des données réelles.

B.3 Structure du code

Le code Java développé pour ce travail est organisé en quatre packages principaux, chacun jouant un rôle spécifique dans l'architecture logicielle. Nous présentons ci-dessous un aperçu de leurs fonctions respectives.

Package `gnuplotOut` : Ce package fait office d'interface avec le logiciel de visualisation Gnuplot. Il permet d'exporter les valeurs mesurées pour différentes métriques vers des fichiers, ainsi que de générer automa-

tiquement les scripts Gnuplot nécessaires à la production de graphiques à la fin des simulations. Ces scripts peuvent ensuite être modifiés manuellement si besoin.

Package `configuration` : Ce package contient l'infrastructure permettant de gérer les configurations des simulations et le paramétrage du traçage des métriques. Le système de configuration a été conçu de manière générique et extensible, notamment grâce à l'utilisation de la réflexivité en Java. La classe centrale `Configuration` permet de charger dynamiquement tout un ensemble d'entrées de la forme `clé=valeur`, que ce soit à partir d'un fichier (une entrée par ligne) ou d'un tableau (arguments du programme Java). La spécialisation de cette classe permet de définir des attributs qui seront automatiquement initialisés lors du chargement des paramètres grâce à la correspondance entre la clé d'une entrée et le nom d'un attribut.

Package `sandPileModels` : Ce package regroupe l'ensemble des modèles du tas de sable et leurs variantes présentées dans ce manuscrit. La classe `SandPileModel`, conçue pour être étendue, constitue la base de tous les modèles implémentés. Le package inclut également les différentes stratégies d'éboulement (`ToppleStrategy`), définissant la façon dont les grains sont redistribués aux voisines d'une cellule lors de son effondrement. Plusieurs stratégies sont disponibles dans la classe utilitaire `ToppleStrategies`. Le comportement utilisé dans les expériences présentées repose sur la stratégie `ToppleStrategies.DEFAULT`, qui distribue un grain par voisine.

Package `simulation` : Ce package centralise tout ce qui a trait à la conduite des simulations expérimentales. Il fournit des simulateurs permettant de tester les modèles et de mesurer les différentes métriques. Par ailleurs, il propose une architecture générique pour la gestion des métriques via la classe `Metric<T>`, incluant le stockage de leurs mesures, leur agrégation et leur export pour traçage via le package `gnuplotOut`.

B.4 Utilisation du code

Dans cette section, nous verrons comment utiliser les programmes proposés pour mener à bien les expériences. Avant de pouvoir utiliser le code, il est nécessaire de préparer l'environnement d'exécution ainsi que les fichiers de configuration, en fonction du programme que l'on souhaite exécuter. Nous verrons également comment utiliser les scripts Bash fournis, qui permettent d'automatiser le lancement de séries d'expériences (batchs).

B.4.1 Préparation de l'environnement

Tout d'abord, l'exécution du programme nécessite que Java, en version 21 ou supérieure, soit installé sur la machine. Le projet étant géré avec Maven, ce gestionnaire de projet doit également être installé. Maven permet notamment de gérer automatiquement les dépendances externes, dont GraphStream, une bibliothèque utilisée pour construire les diverses structures sur lesquelles les modèles de tas de sable (canonique ou dérivés) sont exécutés. GraphStream peut être consultée à l'adresse suivante : https://graphstream-project.org/.

L'exécution du projet avec Maven se fait par la commande suivante :

```
mvn exec:java -Dexec.mainClass="[programme]" -q
```

Il faut remplacer `[programme]` par le nom complet de la classe exécutable. Par exemple, le programme de simulation du tas de sable canonique est :

```
litis.ri2c.heleine.paulin.selfAdaptiveSandSieve.SimulationSandPile
```

Tous les programmes mis à disposition ont la même base de nom. Seul le suffixe change selon le programme désiré. Si le passage des paramètres se fait via les arguments, il faut ajouter l'option suivante :

```
-Dexec.args="param1=valeur1 param2=valeur2"
```

La spécification des paramètres est abordée en détails dans la section suivante.

B.4.2 Les programmes et leurs paramètres

Plusieurs programmes exécutables sont mis à disposition à la racine `selfAdaptiveSandSieve`. Cinq nous intéressent ici, dont un récapitulatif est proposé dans le Tableau B.1. Ils partagent tous un même ensemble de paramètres de base regroupés dans le Tableau B.2. Chaque entrée de configuration prend la forme `clé=valeur`.

Programme	Description
`SimulationSandPile`	Simulation du tas de sable canonique dans différentes topologies
`SimulationNaiveLESASS` `SimulationProportionalLESASS`	Simulation du tamis auto-adaptatif basé sur l'entropie locale
`SimulationGossipSASS`	Simulation du tamis auto-adaptatif basé sur le protocole de bavardage
`GraphExamples`	Génération des visuels de l'étude illustrative du recâblage et de la dégradation

TABLE B.1 – Récapitulatifs des programmes mis à disposition.

B.4.2.1 Programme de simulation du tas de sable canonique

Le programme `SimulationSandPile` permet de simuler le tas de sable canonique sur différentes topologies. Il est notamment utilisé pour mener les expériences du Chapitre 4, consacrées à l'étude de la robustesse du modèle.

Ce programme simule le tas de sable sur des structures potentiellement recâblées et/ou dégradées, selon les paramètres définis. Les paramètres utilisés sont ceux décrits dans le Tableau B.2, auxquels s'ajoute un paramètre spécifique : `uniqueRewire=[Integer]`. Il permet de ne simuler qu'un seul taux de recâblage à la fois, tout en conservant une arborescence de fichiers de sortie identique à celle d'une exécution multi-taux (par exemple 0-100). Il est particulièrement utile pour les expériences à recâblage fin (par paliers de 1%), sans dégradation (résultats Section 4.1.1), afin de limiter la consommation mémoire et les ressources en threads.

Clé	Type de valeur	Définition
`gridSize`	Integer	Taille d'un côté de la grille carrée
`neighborhood`	String	Nom du voisinage utilisé pour générer la grille : `vonneumann` ou `moore`
`cycles`	Integer	Nombre de cycles de la simulation
`useInitCycles`	Boolean	Définit si 10% de cycles d'initialisation (exempts de mesures) sont ajoutés au début de la simulation ou non
`toppleStrategy`	String	Nom de la stratégie d'éboulement des cellules utilisée : `default`
`dynamicThreshold`	Boolean	Définit si le seuil critique dynamique (5.2) sera utilisé ou non
`simultaneousThreads`	Integer	Nombre max. d'expériences exécutées en parallèle
`seeds`	Integer	Nombre d'expériences indépendantes simulées et moyennées
`initialRandomSeed`	Long	Graine aléatoire initiale définissant l'aléatoire de toutes les expériences
`rewireStart`	Integer	Valeur de 0 à 100 représentant le pourcentage de recâblage initial de la grille
`rewireStop`	Integer	Valeur de 0 à 100 représentant le pourcentage de recâblage de la grille maximal qui sera fait
`rewireStep`	Integer	Finesse de la progression du recâblage de `rewireStart` à `rewireStop`
`removeRate`	Integer	Valeur de 0 à 100 représentant le pourcentage de cellules supprimées durant la première phase du processus de dégradation (Section 4.1.2)

TABLE B.2 – Paramètres généraux des programmes de simulation.

Les paramètres peuvent être spécifiés directement en ligne de commande lors de l'exécution du programme, ou bien listés dans un fichier de configuration, `configs/simulationConfigs/sandpile.config` (depuis la racine du dépôt), qui sera utilisé par défaut en l'absence d'arguments.

Un script Bash, `sandPile_simulations.sh`, situé à la racine du projet, permet de lancer l'ensemble des expériences portant sur la robustesse du tas de sable. Il est à noter que les simulations sont longues (plusieurs jours) avec les paramètres par défaut, en raison du volume d'expériences (25 expériences simultanées pour chaque couple recâblage-dégradation).

B.4.2.2 Paramètres des politiques de taille des grains du tamis

De nouveaux paramètres, spécifiques au fonctionnement du tamis, viennent s'ajouter à ceux présentés dans le Tableau B.2. Ils concernent la politique d'évolution de la taille des grains injectés dans le tamis au cours du temps.

Cinq politiques distinctes sont proposées, chacune disposant de ses propres paramètres. Les clés associées à ces paramètres sont systématiquement préfixées par `grainsSizePolicy`, afin d'en faciliter l'identification et l'utilisation. Par exemple : `grainsSizePolicy.name=constant`. L'ensemble des paramètres relatifs à ces politiques est détaillé dans le Tableau B.3.

Taille constante		
Clé	**Type de valeur**	**Définition**
`name`	String	Nom de la politique à utiliser : `constant`
`constSize`	Integer	Taille constante des grains au fil de la simulation
Taille sinusoïdale		
Clé	**Type de valeur**	**Définition**
`name`	String	Nom de la politique à utiliser : `sinusoid`
`start`	Integer	Valeur de départ de la sinusoïde (moyenne de la fonction)
`amplitude`	Integer	Fluctuation de la taille autour de `start`
`period`	Integer	Nombre de cycle que dure une vague complète de la sinusoïde
Taille incrémentale		
Clé	**Type de valeur**	**Définition**
`name`	String	Nom de la politique à utiliser : `incremental`
`step`	Integer	Valeur de départ et d'incrémentation
`every`	Integer	Nombre de cycle avant une incrémentation
Taille aléatoire avec pics		
Clé	**Type de valeur**	**Définition**
`name`	String	Nom de la politique à utiliser : `random-spike`
`gaussianAvg`	Integer	Valeur moyenne de la base aléatoire
`gaussianStdDev`	Integer	Écart-type des valeurs de la base aléatoire
`pSpike`	Double	Probabilité de déclenchement d'un pic à chaque cycle
`spikeDurationMin`	Integer	Durée minimum d'un pic de charge
`spikeDurationMax`	Integer	Durée maximum d'un pic de charge
`spikeMin`	Integer	Taille de grain minimum durant un pic
`spikeMax`	Integer	Taille de grain maximum durant un pic
Taille réelle		
Clé	**Type de valeur**	**Définition**
`name`	String	Nom de la politique à utiliser : `dataset`
`datasetName`	String	Nom du dataset : `auvergrid`, `nordugrid` ou `sharcnet`
`startTime`	Long	Date en secondes du début de simulation dans la trace : - AuverGrid : `1143849600L` (1er avril 2006) - NorduGrid : `1136073600L` (1er janvier 2006) - SHARCNET : `1157068800L` (1er septembre 2006)
`maxArrivalCycle`	Long	Durée en secondes observée dans la trace

TABLE B.3 – Paramètres des politiques de taille de grains.

B.4.2.3 Programmes de simulation du tamis auto-adaptatif : entropie locale

Les programmes `SimulationNaiveLESASS` et `SimulationProportionalLESASS` permettent de simuler un tamis auto-adaptatif basé respectivement sur la méthode par entropie locale naïve (voir Section 5.4.2) et sur la méthode proportionnelle (voir Section 5.4.3). La majorité des paramètres décrits précédem-

ment dans les Tableaux B.2 et B.3 sont à utiliser pour ces simulations. À noter que les paramètres relatifs au recâblage et à la dégradation ne sont pas utilisés dans ce cadre. Deux paramètres supplémentaires, spécifiques à l'adaptation par entropie locale, doivent être fournis conformément au Tableau B.4.

Clé	Type de valeur	Définition
`entropyWindow`	Integer	Taille de la fenêtre d'entropie utilisée pour calculer l'entropie locale (E_{locale}) des cellules
`entropyRef`	Double	Entropie de référence (E_{ref}) utilisée pour prendre les décisions d'adaptation des cellules

TABLE B.4 – Paramètres spécifique aux méthodes par entropie locale.

Les paramètres peuvent être spécifiés directement en ligne de commande lors de l'exécution, ou bien placés dans un fichier de configuration situé à l'emplacement suivant, relatif à la racine du dépôt :

```
/configs/simulationConfigs/sandsieve.config
```

Ce fichier est utilisé par défaut si aucun argument n'est fourni.

Enfin, un script Bash `lesass_simulations.sh`, situé à la racine du projet, permet de lancer automatiquement les expériences du Chapitre 5. Ce script prend en paramètre le nom du programme à exécuter : `SimulationNaiveLESASS` ou `SimulationProportionalLESASS`. Il est important de noter que ce script ne permet de lancer qu'une seule politique de taille des grains à la fois. Pour modifier la politique utilisée, il faut éditer le script en commentant la ligne d'arguments de la politique désactivée, et en décommentant celle de la politique souhaitée.

B.4.2.4 Programme de simulation du tamis auto-adaptatif : protocole de bavardage

Le programme `SimulationGossipSASS` permet de simuler un tamis auto-adaptatif basé sur le protocole de bavardage, tel que présenté dans la Section 5.5. Ce programme exploite les paramètres généraux listés dans les Tableaux B.2 et B.3, à l'exception de ceux relatifs au recâblage et à la dégradation, qui ne sont pas pris en compte. Aucun paramètre spécifique supplémentaire n'est requis pour ce modèle.

Les paramètres peuvent être spécifiés directement en ligne de commande lors de l'exécution, ou bien placés dans un fichier de configuration situé à l'emplacement suivant, relatif à la racine du dépôt :

```
/configs/simulationConfigs/sandsieve.config
```

Ce fichier est utilisé par défaut si aucun argument n'est fourni.

Étant donné qu'il n'existe pas de variations de paramètres propres à ce modèle à explorer, aucun script Bash dédié à l'exécution en batch n'est fourni. L'exécution du programme se fait donc manuellement, via ligne de commande ou à l'aide d'un fichier de configuration adapté.

B.4.2.5 Génération des illustrations du recâblage et de la dégradation

Le programme `GraphExamples` est utilisé pour générer les visuels présentés dans l'étude illustrative du recâblage et de la dégradation (Section 4.3). Il produit des représentations visuelles d'une grille régulière de

taille 16, modélisée à l'aide de la bibliothèque GraphStream. Les visuels sont générés pour différents niveaux de dégradation (suppression de nœuds), allant de 0% à 90% par paliers de 5%, et pour des taux de recâblage allant de 0% à 100% par paliers de 10%. Le programme ne nécessite aucun paramètre d'entrée et peut être exécuté directement.