UNIVERSITÉ DU QUÉBEC À MONTRÉAL

DISPONIBILITÉ DES DONNÉES DANS LES CENTRES DE DONNÉES Á CARACTÉRISTIQUES HÉTÉROGÈNES

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

MOUHAMAD DIEYE

DÉCEMBRE 2016

REMERCIEMENTS

Je tiens à exprimer de sincères remerciements ainsi que toute ma gratitude à mes encadrants Pr. Halima Elbiaze et Pr. Mohamed Faten Zhani. Tout au long de l'élaboration de ce mémoire, j'ai eu l'honneur de bénéficier de leur soutien scientifique, moral et financier. Je tiens à leur exprimer ma reconnaissance pour leur disponibilité ainsi que leurs conseils avisés.

Je remercie tous mes collègues du laboratoire TRIME à l'UQAM. Je citerais Wassim S. Atoui, Amina Mseddi, Mlika Zoubeir et Fatoumata Baldé pour leur assistance.

Je réitère mes remerciements et respects envers mes très chers parents, M. Abdoulaye Dieye et Mme Fatoumata Hawra Diop, pour leur soutien indéfectible et inconditionnel. J'espère qu'ils retrouveront à travers ce travail, un gage de l'estime et de la gratitude que je leur porte en mon coeur.

Je voudrais, en particulier, exprimer ma sincère gratitude à M. Serigne Mor Diop pour son indéfectible soutien moral ainsi que ses encouragements. Au delà de ses prières précieuses, j'admire sa capacité d'écoute, de modestie et d'ouverture d'esprit. Je citerais également M. Oumar Faroukh Diop pour son affection et sa gentillesse à mon égard.

Je remercie également mes grands-parents Mme Sokhna Fatou Ndiaye, Feu M. Assane Dieye et Feue Mme Sokhna Fatou Diop pour leurs prières ainsi que leurs encouragements tout au fil des années.

Je dédie ce travail à mes frères et soeurs Ibrahim, Khadija, Ahmad Tijani, Al

Hadji Malick, Assane, Mariétou et Mor.

Pour leur assistance et encouragements, j'adresse également ma gratitude à M. Babacar Matar Ndiaye, Mme Fagaye Diop, M. Momar Sarr Sall, , M. Saiba Sylla et M. Blaise Turpin.

Je n'oublie pas de remercier tous mes enseignants à l'Université du Québec à Montréal pour le savoir-faire qu'ils m'ont généreusement inculqué.

À toute ma famille, mes amis, tous ceux qui, de près ou de loin, m'ont aidé dans la réalisation de ce travail, je vous exprime de sincères remerciements.

Mouhamad Dieye

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

RÉSUMÉ

Les services de stockage infonuagique (*cloud storage*) sont devenus, à l'ère du Big Data, le choix standard en matière de stockage de données notamment grâce à leur rentabilité ainsi qu'à l'apparente capacité de stockage illimité qu'ils offrent. En raison du succès, toujours aussi croissant de ces services, les fournisseurs de stockage infonuagique sont de plus en plus incités à améliorer davantage leurs infrastructures de stockage afin de garantir des conditions strictes de disponibilité ainsi qu'un temps de réponse adéquat suite à des requêtes d'accès aux données stockées. D'autant plus que des études récentes ont globalement estimé le coût associé à l'arrêt d'applications critiques d'entreprise (*critical business applications*) entre $84,000 et $108,000 par heure.

En dépit de nombreux efforts visant à préserver la disponibilité des données dans les systèmes de stockage infonuagique, les solutions existantes ont souvent tendance à négliger l'aspect hétérogène de l'infrastructure de stockage sous-jacente en termes de capacité, vitesse d'E/S, taux de pannes, etc. Naturellement, nous nous posons la question à savoir : quel est l'impact de l'hétérogénéité sur la capacité d'un fournisseur de stockage infonuagique à satisfaire les exigences en termes de disponibilité ? Et le cas échéant, quelle stratégie de gestion de données devrait être adoptée pour faire face au défi qu'apporte l'hétérogénéité dans l'accommodement des contraintes en termes de disponibilité ?

Ainsi, dans ce mémoire, nous présentons une plateforme de gestion de données qui prend en considération différents paramètres des disques durs ayant une influence sur la satisfaction de Garantie de Niveau de Service (*Service Level Agreement - SLA*) en termes de temps d'accès et de disponibilité des données tout en garantissant une surcharge (*overhead*) minime en termes de stockage, de migration de données et de consommation d'énergie.

Nous nommons ce nouveau mécanisme de gestion de réplicas de données, Heron. Il se base sur un algorithme génétique auquel nous combinons des techniques de prévision de la demande et des pannes de disque durs pour ajuster, de manière autonome, le nombre et l'emplacement des réplicas de données de sorte à garantir la disponibilité des données tout en tenant compte des fluctuations de performance dues à l'hétérogénéité des équipements de stockage et du trafic.

À travers des simulations réalistes, nous constatons qu'Heron, suivant plusieurs

scénarios de simulation et en comparaison à d'autres solutions ne tenant pas compte de l'hétérogénéité, présente en moyenne une différence de 4% pour le taux de violation en termes de disponibilité et de temps de réponse. En outre, Heron réduit considérablement la quantité de réplicas créés et migrés au fil du temps ainsi que l'énergie totale consommée. Enfin, nous montrons également qu'Heron induit une pénalité globale significativement moindre, suite aux violations en termes de disponibilité.

Mots clés : Systèmes de stockage infonuagiques, disponibilité des données, tolérance aux pannes, algorithmes génétiques

CHAPITRE I

INTRODUCTION

1.1 Mise en contexte

La création et le partage de l'information s'apparentent, de nos jours, à un phénomène banal d'un monde ultra connecté à travers Internet. En effet, de façon quotidienne, un nombre considérable d'individus s'instruisent, travaillent, communiquent à travers les réseaux sociaux, etc. Puis, s'ajoute à cela qu'au cours de ces deux dernières décennies, en plus de la multiplication d'équipements générateurs de contenus (texte, audio et vidéo), nous assistons à une vulgarisation des moyens de stockage des données ainsi qu'à une modernisation des réseaux de communications avec comme résultat, une explosion de données devenue aujourd'hui la source de nombreux défis faisant l'objet de sérieuses études (Somasundaram et Shrivastava, 2009; Beath *et al.*, 2012; Milligan et Selkirk, 2002).

D'ailleurs, selon une étude réalisée par Gartner (Adams et Mishra, 2010), on estime qu'au niveau des centres de données, le défi majeur à surmonter consiste à gérer convenablement le volume exponentiel de données, cela bien avant les problématiques liées à la gestion de la performance et d'évolutivité du système. C'est sans doute ce qui explique, que dès 2011, plus de 62% des grandes entreprises prévoyaient, dans un futur proche, d'investir et de migrer leurs activités vers une solution de stockage massif dans l'infonuagique (Adams et Mishra, 2010).

1.2 Motivation

On peut facilement comprendre cet engouement par rapport aux solutions de stockage infonuagique car la rentabilité économique et la compétitivité des entreprises sont aujourd'hui devenues fortement dépendantes, dans un environnement concurrentiel sans cesse plus acerbe, de leur capacité à accéder et traiter de façon fiable et efficace leur données. Autrement dit, la disponibilité des données est devenue un aspect essentiel à l'essor économique et au survie même des activités de l'entreprise (Doughty, 2000).

La disponibilité est définie comme étant la probabilité qu'un système soit opérationnel à chaque fois que son utilisation est requise (Elsayed, 1996), représentant de ce fait la proportion du temps pour laquelle un système est en état de marche.

Pour témoigner de l'importance de la disponibilité des données pour les entreprises, une étude (ITG, 2016) indique que 87% des entreprises ayant subi, pour une quelconque raison, une perte d'accès à leur données pendant plus d'une semaine sont inévitablement tombées en faillite dans l'année qui suit. De même, une étude récente (Arnold, 2010) estime globalement le coût associé à l'arrêt d'un service critique entre 84 000$ et 108 000$ par heure. Pire encore, on estime que pour 25% des entreprises, les pertes dépassent plus de 500 000$ par heure (Sombers, 2007) sans compter que selon Dunn et Bradstreet, plus de 59% des entreprises du Fortune 500 subissent une indisponibilité de service d'au moins 1,6 heures chaque semaine; c'est-à-dire, l'équivalent d'une perte annuelle de 46 000 000$ (Arnold, 2010).

C'est donc sans surprise que les entreprises exigent désormais un respect strict de conditions d'accès aux données par l'intermédiaire de contrats de garanties de qualité de service (*service level agreements - SLA*) dans lequel la disponibilité occupe une place fondamentale du fait qu'une indisponibilité des données peut avoir

des conséquences désastreuses, notamment une interruption de service, entraînant des pertes de revenues auxquelles s'ajoutent d'énormes dommages en termes de réputation. Tout cela se traduit potentiellement en une chute des prix des actions de l'entreprise, en pertes d'utilisateurs, etc. (Somasundaram et Shrivastava, 2009; Beecher *et al.*, 2013).

Au-delà des préoccupations relatives à la disponibilité des données, les défis liés à la gestion et au traitement de données dans un centre de stockage infonuagique sont également grandement amplifiés par des considérations légales, réglementaires et contractuelles (Somasundaram et Shrivastava, 2009). À titre d'exemple, les institutions bancaires sont tenues par un cadre légale et réglementaire de maintenir de manière fiable et intègre leurs données sur une longue période d'années (Somasundaram et Shrivastava, 2009; Abadi, 2009) tout en maintenant des impératifs de qualités de service rigoureuse. En plus de cela, elles (les institutions bancaires) sont en possession de données plus ou moins sensibles limitant géographiquement les possibilités de stockage de données pour des raisons géopolitiques et géoéconomiques. Ces contraintes se voient alors répercutées aux opérateurs de stockage infonuagique à qui on interdit de stocker/répliquer certains types de données dans un centre de données situé dans un pays étranger. De même, on pourrait exiger d'un opérateur de stockage infonuagique qu'il conserve des données bien au-delà de la date d'accès final par le propriétaire des données occupant ainsi de l'espace de stockage. Tout cela, bien sûr, rend particulièrement difficile la tâche consistant à garantir une disponibilité convenable des données.

C'est donc dans ce contexte que les solutions de stockage infonuagique se sont imposées comme étant un paradigme de premier plan pour le stockage et la livraison des données grâce à leur souplesse, en permettant effectivement de mutualiser et de simplifier les opérations informatiques grâce à la virtualisation. De même, les systèmes de stockage infonuagique représentent, pour les entreprises, une so-

lution rentable à la fois d'un point de vue économique et opérationnelle grâce à son modèle de paiement à l'usage des ressources tout en permettant de déléguer les complexités opérationnelles liées à la gestion des données à un opérateur de stockage infonuagique (*cloud storage provider - CSP*).

Dès lors, les CSP se voient confier à travers un SLA, la lourde tâche de garantir une disponibilité adéquate des données stockées. Dans cette perspective, les CSPs ont typiquement recours, entre autres techniques, à l'utilisation de la réplication de données comme un moyen efficace d'assurer les exigences SLA, notamment en termes de disponibilité et de temps d'accès aux données dans leur systèmes. À titre d'exemple, on peut citer Amazon S3, GFS (Ghemawat *et al.*, 2003), HDFS (Shvachko *et al.*, 2010), etc.

La réplication de données est une technique qui permet, à travers une démultiplication des sources d'une donnée, d'assurer la continuité des services dans un environnement prompt à de multiples et diverse défaillances (matérielles, logicielles et humaines). Étant l'une des stratégies les plus utilisées dans un centre de données pour garantir la disponibilité (Con, 2014), le mécanisme de réplication de données a naturellement fait l'objet de nombreuse études dont l'objectif est d'accroître la disponibilité et la fiabilité de systèmes complexes (Sun *et al.*, 2012; Wei *et al.*, 2010; Lin *et al.*, 2013a; Abad *et al.*, 2011; Li *et al.*, 2012; Goel et Buyya, 2007). Les réplicas de données sont alors employés comme moyen de récupération des données en cas de pannes, plus particulièrement des disques durs, préservant de ce fait, la disponibilité des données. Ils peuvent, également, servir à alléger la charge du trafic destiné vers certains disques durs avec pour objectif de réduire le temps d'accès aux données stockés.

Dans ce cadre, le principal défi pour un CSP par rapport à la réplication de données consiste à déterminer, pour chaque bloc de données, le nombre et l'emplacement

optimal des réplicas nécessaire à la satisfaction des exigences édictées par les SLAs.

1.3 Problématique

En dépit du fait que la réplication de données représente une technique efficace pour préserver la disponibilité des données dans un environnement propice aux défaillances, la réplication de données pose, dans le contexte du stockage infonuagique, de nombreuses problématiques pour les CSP qui auront la lourde tâche d'apporter des solutions efficaces et rentables.

D'une façon générale, les données en provenance des clients du système de stockage infonuagique ont des caractéristiques d'importance extrêmement variées, non seulement par rapport au fonctionnement des services des clients infonuagiques mais également des contraintes légales, réglementaires, de temps d'accès, etc.

À travers cette étude, nous voulons mettre en exergue plusieurs problématiques du mécanisme de réplication de données dans le cadre d'infrastructures de stockage infonuagique, et qui posent un obstacle à l'accomplissement de l'objectif consistant à garantir les exigences SLA tout en considérant un aspect largement ignoré par les recherches actuelles, à savoir, l'hétérogénéité des équipements de stockage.

La première problématique est que les pannes d'équipements constituent une occurrence plus ou moins fréquente en raison de la grande échelle de l'infrastructure. Les pannes vont des problèmes d'énergies, des pannes de disques durs, en passant par des interruptions réseaux jusqu'à l'erreur humaine. Autant d'aspects qui ont des effets non négligeables et qui peuvent provoquer une indisponibilité ou, pire, des pertes de données (Bonvin *et al.*, 2010; Pinheiro *et al.*, 2007) même en présence de réplicas de données.

Dans le cadre de ce travail, nous focalisons notre attention sur les pannes de

disques durs ainsi que leurs effets sur la disponibilité des données d'autant plus que de nombreuses études (Pinheiro *et al.*, 2007; Schroeder et Gibson, 2007; Vishwanath et Nagappan, 2010; Sankar *et al.*, 2013) concluent qu'ils sont à l'origine des pannes dans les centre de données (plus de 71% selon (Sankar *et al.*, 2013)). Une panne de disque dur a, par conséquent, une influence cruciale sur la disponibilité des données puisque la disponibilité d'une donnée dépend fortement de la disponibilité du disque dur le contenant.

Au delà d'être la cause principale de l'indisponibilité des données, le comportement pré-panne (*failure behaviour*) des disques demeure également très complexe. C'est dans ce cadre que des travaux récents (Schroeder et Gibson, 2007; Pinheiro *et al.*, 2007; Schroeder et Gibson, 2010) ont, en particulier, mis l'accent sur le fait qu'en réalité, les disques durs éprouvent des variations en termes de taux de pannes (*failure rate*) au fil du temps selon divers paramètres tels que l'âge, le taux d'utilisation, le modèle, etc. Cela entre nettement en contradiction avec l'hypothèse assez répandue selon laquelle le taux de panne des disques durs peut être considéré comme étant constant ou encore modélisable selon une courbe baignoire (*bathtub curve*). Si bien que modéliser la disponibilité ou encore la fiabilité d'un disque dur suivant suivant une distribution exponentielle des pannes de disques devient invalide puisque le taux de pannes est variable au fil du temps. D'ailleurs, sur ce point, une étude (Schroeder et Gibson, 2007) dénote qu'il existe souvent un écart significatif (un facteur de 6 à 30) entre les valeurs de taux de pannes constantes retenues et celles observées en réalité. Dans ces conditions, considérer une distribution exponentielle pourrait aboutir à quantifier de manière erronée la disponibilité (Sun et Han, 2001; Elerath, 2000; Shah et Elerath, 2005), entraînant de ce fait de mauvaises décisions pour garantir la disponibilité des données dans le cadre de la réplication de données.

Au-delà de la variation du taux de pannes au fil du temps, il existe évidemment

une hétérogénéité inhérente au niveau même des disques durs. En effet, il existe typiquement, au sein d'une infrastructure de stockage infonuagique, des milliers de serveurs contenant des disques durs dont les caractéristiques varient notamment en termes de capacité, d'âge, de vitesse d'E/S (c-à-d entrée/sortie), de consommation énergétique, de modèle, etc. (Boutaba *et al.*, 2011; Backblaze, 2015).

Cette hétérogénéité des caractéristiques est confirmée par l'opérateur Backblaze dans son analyse de près de 50000 disques durs au sein d'un infrastructure de stockage infonuagique réel. Effectivement, les disques durs provenaient de plus d'une quinzaine de modèles avec des caractéristiques toutes aussi diverses. Par exemple, selon le modèle considéré, la capacité de stockage varie entre 1 à 4 TB, la puissance de consommation énergétique active entre 4.4 à 12 W, etc. (Backblaze, 2015).

En outre, (Sharma *et al.*, 2011) expliquent qu'une infrastructure de stockage homogène est pratiquement impossible d'un point de vue économique au-delà du fait que cela rendrait le CSP trop dépendant d'un unique fabricant.

Dès lors, si l'on tient compte de l'hétérogénéité, un certain nombre d'observations pertinentes se dégagent et doivent être prises en considération dans le cadre de la gestion de données dans un centre de données.

En premier lieu, négliger l'hétérogénéité des taux de pannes ainsi que leur variation au fil du temps peut entraîner une mauvaise estimation de la disponibilité des données au fil du temps. Pour cette raison, le nombre de réplicas ainsi que leurs emplacements doivent, idéalement, non seulement tenir compte de la disponibilité même des disques durs récipiendaires mais également, pouvoir être dynamiquement réajustés (c-à-d le nombre et les emplacements des réplicas) au gré des variations de taux de pannes des disques durs.

Un autre aspect à tenir en compte concerne la variabilité de la demande. En ce sens que les centres de données reçoivent typiquement des requêtes d'accès à diverses ressources avec des niveaux de demande et d'exigences en termes de performance tout aussi variés (Zhang *et al.*, 2013; Zhang *et al.*, 2011; Sharma *et al.*, 2011; Lin *et al.*, 2013b; Boutaba *et al.*, 2011). À partir de là, le CSP se retrouve régulièrement confronté à un dilemme selon lequel, il doit obligatoirement traiter les requêtes d'accès aux données d'utilisateurs de manière à conjointement satisfaire les impératifs des clients (c-à-d rapidité de réponse, bas coût de traitement, etc.) ainsi que ceux des CSP (c-à-d efficience des ressources et de la consommation d'énergie, coûts de traitement pour les opérations, etc.) (Delimitrou *et al.*, 2011; Delimitrou et Kozyrakis, 2013).

D'autant plus que le temps de réponse suite aux requêtes d'accès aux données a non seulement un impact direct sur la qualité de service des utilisateurs (Bauer et Adams, 2012) mais également un impact économique non négligeable. À ce propos, d'après (Linden, 2006), on observe une réduction de trafic de l'ordre de 20% pour Google.com lorsque le temps de réponse dépasse plus de 500 ms. De façon similaire, pour 100 ms en plus, le nombre d'achats sur Amazon.com diminue de 1%.

Il est donc essentiel, dans ce contexte, que les requêtes d'accès aux données clients soient redirigées vers les emplacements contenant un réplica en tenant compte de la réalité liée à l'hétérogénéité de leur capacité de traitement. En effet, un modèle de gestion des données qui ne tient pas compte de l'hétérogénéité de performance des disques durs pourrait certes distribuer les requêtes d'accès au niveau des emplacements des réplicas de manière équitable. Toutefois, cette stratégie pose problème dans la mesure où certains disques durs, ayant des vitesse de traitement E/S différentes, peuvent ne pas être capables de traiter convenablement les requêtes dans les délais requis, entraînant ainsi des violations d'exigences SLA.

Une dernière considération pour une stratégie de gestion de données dans un centre de données hétérogène consiste à prendre en compte les coûts liés à la gestion des données notamment au cours du processus de création, de migration et de suppression de réplicas de données. Dans le cadre de centre de données où l'infrastructure de stockage sous-jacente se caractérise par une hétérogénéité et une variabilité prononcées, assurer une stabilité des configurations afin d'éviter des opérations fréquentes de création et de migration de réplicas devient une nécessité étant donné qu'un réplica se doit d'être une copie exacte d'une donnée en tout temps, nécessitant donc des opérations de synchronisation entre tous les réplicas. Autrement dit, toute opération de gestion effectuée sur un réplica doit être répercutée au niveau des réplicas restants, impliquant, de ce fait, un échange potentiellement important de données entre emplacements de réplicas qui peut potentiellement être à l'origine de dégradations de performances réseau en termes de latence ainsi qu'une augmentation de la consommation de bande passante et de l'énergie. En outre, il est important de noter que, typiquement, les réplicas nouvellement migrés ou créés ne sont accessibles qu'une fois la synchronisation complète (Somasundaram et Shrivastava, 2009). Dès lors, dépendant de la taille des réplicas et en fonction des performances à la fois des disques durs impliqués et du réseau dans sa globalité, les opérations de gestion peuvent alors prendre des durées significatives durant lesquelles les conditions édictées par le SLA ne sont plus respectées.

Pour ces raisons, nous estimons qu'un système de gestion de réplicas dans un contexte de centre de données à infrastructure de stockage hétérogène doit être pro-active. En ce sens qu'il doit utiliser des outils de prédictions de la demande et de pannes de disques durs afin de tirer des tendances significatives permettant de décider la stratégie optimale à appliquer, en fonction des variations de la demande et de la disponibilité des disques durs. Par exemple, faudrait-il constamment ré-

ajuster le nombre et l'emplacement des réplicas pour garantir la disponibilité ainsi qu'un temps de réponse adéquat ou encore maintenir un nombre excédent de réplicas, en dépit d'une consommation énergétique accrue afin de maintenir la stabilité du système lorsque l'on détecte une recrudescence de la demande dans un futur proche.

Tout au long de ce mémoire, nous nous évertuerons à répondre à ces interrogations.

1.4 Méthodologie

Dans le but d'évaluer la pertinence de notre plateforme de gestion de données dans un système stockage infonuagique hétérogène, nous définissons tout d'abord le fonctionnement de notre plateforme au sein de laquelle nous décrivons la manière dont les données sont, dans un premier temps, stockées et puis récupérées à travers des requêtes d'accès émises par les clients du système de stockage.

Nous formulons ensuite analytiquement la problématique de la gestion des réplicas dans une infrastructure de stockage infonuagique hétérogène par un problème d'optimisation dont l'objectif est de minimiser les coûts de gestion des réplicas tout en portant une attention particulière à respecter les contraintes de disponibilité et de temps de réponse suite à une requête d'accès. On notera que dans le cadre de cette étude, nous partons de l'idée qu'il existe deux considérations importantes à prendre en compte. La première est que la sévérité des exigences en termes de disponibilité dénote également l'importance de ces données aux yeux de l'utilisateur propriétaire des données. La deuxième concerne la stabilité globale du système. Ainsi, la solution proposée doit à la fois garantir que les données ayant des exigences SLA plus strictes soient satisfaites en priorité mais qu'également, elle (la solution) puisse sélectionner un schéma d'allocation de réplicas (c'est à dire, les emplacements et le nombre de réplicas pour une donnée) tendant vers une solution proche de l'optimal et stable au fil du temps en dépit du fait qu'elle

ne l'est pas au présent.

Une analyse de la complexité est ensuite faite afin de déterminer notamment la faisabilité de la problématique posée. À la suite de cela, nous présentons une heuristique basée sur un algorithme génétique avec pour objectif de trouver une stratégie de gestion de réplicas proche de l'optimal pour chaque donnée stockée dans l'infrastructure de stockage infonuagique.

La performance de notre solution est évaluée par comparaison avec deux stratégies alternatives : une approche statique et une approche gloutonne. Dans la stratégie statique, nous nous focalisons à fixer le nombre et les emplacements de réplicas tandis que la stratégie gloutonne se préoccupe d'optimiser les emplacements de réplicas. L'objectif de ces comparaisons étant de comprendre l'étendue de l'impact de l'hétérogénéité sur la gestion de données. Autrement dit, jusqu'à quel niveau d'hétérogénéité peut-on se contenter d'influer simplement sur le nombre ou/et les emplacements de réplicas ? Suffirait-il simplement d'ajouter de nouveaux réplicas pour combler les défis qu'apportent l'hétérogénéité des équipements de stockage dans l'infrastructure de stockage infonuagique ?

À notre connaissance, il n'y a pas encore d'études similaires dans la littérature.

1.5 Contribution et organisation du mémoire

L'objectif de ce mémoire est de traiter la problématique de la gestion des données dans un centre de données ayant une infrastructure de stockage hétérogène en prenant en compte les considérations d'hétérogénéité de la demande et des caractéristiques de disques durs, notamment des taux de pannes, de capacité de stockage, de vitesse d'E/S (entrée/sortie), entre autres. Dans ce cadre, nous présentons un plateforme de gestion de réplicas qui exécute en son sein l'algorithme Heron (*HEteROgeNeous-aware replica management*) qui se base sur un algorithme

génétique et qui prend en considération la variation de la charge de trafic et des caractéristiques de disques durs au fil du temps avec comme objectif de satisfaire les exigences SLA en termes de disponibilité et de temps d'accès aux données tout en minimisant la consommation d'énergie. En somme, la contribution de ce mémoire consiste à :

1. Décrire l'état de l'art de la modélisation de la disponibilité dans le contexte de la réplication de données. Nous parlerons des travaux existants ainsi que des caractéristiques pertinentes ayant trait à l'hétérogénéité, la prévision des pannes de disques durs et de trafic et des modèles de gestion des données dans le stockage infonuagique.
2. Formuler une étude analytique tenant compte de l'hétérogénéité afin de satisfaire diverses contraintes de disponibilité et de temps d'accès dans le contexte de la réplication de données dans un centre de données. Nous analysons ensuite la complexité du problème en termes de temps de calcul pour prouver qu'il s'agit d'un problème NP-difficile, dont on conjecture qu'il n'existe pas d'algorithmes exécutables par une machine déterministe en un temps raisonnable.
3. Proposer un nouveau mécanisme de gestion de données que nous appelons Heron, tenant compte des contraintes identifiées lors de l'analyse analytique avant de fournir des résultats d'expérimentation. Nous comparons la solution proposé, Heron, avec deux solutions qui ne tiennent pas compte de l'hétérogénéité des équipements de stockage, notamment en termes de taux de pannes : une approche statique ainsi qu'une approche gloutonne. Ces dernières se différencient par l'aspect (le nombre ou l'emplacement) dont ils se préoccupent principalement.

On notera la publication suivante comme étant pertinente à ce mémoire :

M. Dieye, M. F. Zhani and H. Elbiaze, *"On Achieving High Data Availability in Heterogeneous Cloud Storage Systems"*, In 2017 IFIP/IEEE International Symposium on Integrated Network Management (IM).

Une extension de ce précédent travail est en cours et vise à adapter, dans notre plateforme de gestion de réplicas, plusieurs solutions existantes de gestion de réplicas. De plus, nous tirons avantage, dans ce futur travail, de données fournies par Backblaze, plus récentes et complètes en relation avec les pannes de disques durs.

Le présent mémoire sera organisé comme suit :

Nous commençons par discuter, au chapitre II, des notions relatives au concept de modélisation de la disponibilité. Dans ce contexte, nous détaillons les interactions entre les concepts de fiabilité, de maintenabilité et de disponibilité. Nous différencions alors la disponibilité instantanée de celle asymptotique. À partir de là, nous discutons d'une formulation de la disponibilité instantanée capable de prendre en compte les variations des taux de pannes.

En ayant posé les bases de notre analyse, nous présentons au chapitre III les différents travaux dans la littérature pertinents à notre étude. Notamment, nous parlons des travaux concernant la caractérisation ainsi que les techniques de prévision des pannes de disques durs. Nous concluons ce chapitre en discutant des travaux qui traitent de la réplication de données dans le contexte d'un environnement infonuagique.

Le chapitre IV est consacré à la présentation générale de notre plateforme de gestion. Au chapitre V, nous formulons analytiquement le problème de la gestion des réplicas dans le cadre d'une infrastructure de stockage infonuagique hétérogène

avant de proposer une analyse de la complexité du problème dans laquelle nous établissons que notre problème est NP-difficile, justifiant de ce fait l'utilisation d'une heuristique que nous détaillons également au chapitre V.

Au chapitre VI, nous évaluons les performances de notre solution en la comparant avec une approche statique et une approche gloutonne. Dans ce même chapitre, nous analysons les résultats de cette évaluation. Nous fournissons enfin nos conclusions au chapitre VII.

CHAPITRE II

GÉNÉRALITÉS SUR LA MODÉLISATION DE LA DISPONIBILITÉ

Afin de mieux cerner les contours de notre problématique, nous présentons à travers ce chapitre les propriétés notables ayant trait à la modélisation mathématique ainsi qu'à la gestion de la disponibilité avec comme objectif de pouvoir prendre les meilleures décisions en termes de réplication de données.

2.1 Définition

La disponibilité est définie comme étant la probabilité qu'un système soit opérationnel à chaque fois que son utilisation est requise (Elsayed, 1996). Elle constitue ainsi l'un des indicateurs clés de performance d'un système de stockage infonuagique, indiquant la fréquence à laquelle un système est en état de fonctionnement et permet, en outre, de renseigner sur la rentabilité du système à la fois pour l'opérateur de stockage infonuagique qui cherche à éviter de payer des pénalités suite à une incapacité à satisfaire une exigence SLA mais également pour les clients dont la rentabilité économique dépend de la continuité de leurs services.

Par définition, le concept de disponibilité combine, en son sein, les concepts de fiabilité et de maintenabilité, quantifiant respectivement la durabilité et la quantité de maintenance/réparations envisageable pour la durée de vie du système.

Dans ce cadre, il convient alors de définir au préalable les notions de fiabilité et

de maintenabilité pour mieux comprendre la modélisation de la disponibilité.

2.2 Notions de base

- **Fiabilité**

La fiabilité est la probabilité qu'un produit ou service fonctionne adéquatement et sans panne durant une période de temps spécifiée (Elsayed, 1996).

D'un point de vue mathématique, la fiabilité se définit selon une variable aléatoire $X \geq 0$ dénotant le temps écoulé avant la panne de l'équipement (Elsayed, 1996). La fiabilité à un temps t, notée R(t), s'exprime alors comme une fonction de répartition cumulative définie par :

$$R(t) = P\{X > t\}, \quad t > 0 \tag{2.1}$$

Cette formule représente la probabilité d'un équipement à fonctionner convenablement au-delà du temps t prédéfini.

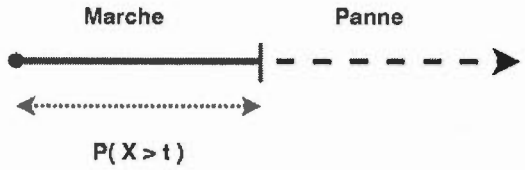

Figure 2.1 Illustration de la fiabilité d'un équipement non réparable.

De manière analogue, l'état de panne d'un équipement se dénote également à travers une fonction de répartition cumulative de pannes $F(t)$, mutuellement exclusive avec $R(t)$ de sorte que :

$$F(t) + R(t) = 1, \quad t > 0 \tag{2.2}$$

On peut, dès lors, définir la densité de probabilité de pannes $f(t)$ et dériver les

formulations suivantes reflétant les relations mathématiques entre la fiabilité et la probabilité du temps avant une panne :

$$R(t) = 1 - F(t) = 1 - \int_0^t f(x)dx \tag{2.3}$$

$$\frac{dR(t)}{dt} = -f(t) \tag{2.4}$$

À partir des relations précédentes, la probabilité qu'un équipement tombe en panne dans un intervalle de temps $[t, t + \Delta t]$ en fonction de la fiabilité $R(t)$ est exprimée par la relation suivante :

$$\int_t^{t+\Delta t} f(t)\ dt = R(t) - R(t + \Delta t) \tag{2.5}$$

Ces relations nous permettent de mettre en évidence la corrélation entre la fiabilité et la fréquence d'occurrences de pannes d'un équipement encore appelée taux de pannes.

- **Taux de pannes**

Le **taux de pannes** $\lambda(t)$ indique, pour l'équipement considérée la fréquence d'occurrence de pannes pour un intervalle de temps $[t,\ t + \Delta t]$ (Elsayed, 1996).

Il se définit alors comme étant la probabilité conditionnelle qu'une panne survienne en une unité de temps dans l'intervalle considéré, avec comme condition qu'une panne ne soit survenue pas antérieurement au temps t :

$$\lambda(t) = \frac{P\{t < X \leq t + \Delta t \mid X > t\}}{\Delta t} \tag{2.6}$$

$$\lambda(t) = \frac{P\{t < X \leq t + \Delta t\}}{\Delta t \cdot P\{X > t\}} \tag{2.7}$$

$$\lambda(t) = \frac{1}{\Delta t \cdot R(t)} \int_t^{t+\Delta t} f(t)\, dt \tag{2.8}$$

$$\lambda(t) = \frac{R(t) - R(t + \Delta t)}{\Delta t\ R(t)} \tag{2.9}$$

Le taux de panne instantané, c'est à dire le taux de pannes à un instant t précis, est déduit à travers la relation suivante :

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{R(t) - R(t + \Delta t)}{\Delta t\ R(t)} = \frac{1}{R(t)} \left[- \frac{dR(t)}{dt} \right] \tag{2.10}$$

Nous établissons alors les relations suivantes entre la fiabilité, le taux de pannes et la probabilité de pannes. Ainsi, en tenant compte de la relation 2.4 nous avons (Elsayed, 1996) :

$$\lambda(t) = \frac{f(t)}{R(t)} = \frac{1}{R(t)} \left[\frac{dF(t)}{dt} \right] \tag{2.11}$$

$$R(t) = e^{-\int_0^t \lambda(x)\ dx} = 1 - \int_0^t \lambda(x)\ dx \tag{2.12}$$

- **Maintenabilité**

La maintenabilité est un concept faisant référence à la capacité de maintenir le fonctionnement ou de réparer l'équipement en accord avec les conditions de fonctionnement et d'intervalle de temps limite définies.

De façon identique à la fiabilité, supposons une variable aléatoire $Y \geq 0$ dénotant le temps nécessaire pour réparer un équipement ou service. Nous définissons alors

la fonction de répartition cumulative du temps de réparation $M(t)$ comme étant :

$$M(t) = P\{Y \leq t\}, \quad t > 0 \tag{2.13}$$

En prenant la densité de probabilité du temps de réparation $m(t)$ d'un équipement dans un intervalle de temps $[t,\ t+\Delta t]$, $m(t)$, nous dérivons les relations suivantes :

$$M(t) = \int_{t}^{t+\Delta t} m(t) \tag{2.14}$$

$$\int_{t}^{t+\Delta t} m(t) = M(t+\Delta t) - M(t) \tag{2.15}$$

Le **taux de réparations** $\mu(t)$ peut se définir comme étant la probabilité qu'un équipement ou service, tombé en panne à partir de t, puisse être réparé avant $t + \Delta t$:

$$\mu(t) = \lim_{\Delta t \to 0} \frac{P\{t < Y \leq t + \Delta t \mid Y > t\}}{\Delta t} = \frac{m(t)}{1 - M(t)} \tag{2.16}$$

Après avoir résumé les concepts fondamentaux de fiabilité, de taux de pannes et de maintenabilité, nous pouvons dès lors discuter de la modélisation de la disponibilité dans ce qui suit.

2.3 Modélisation de la disponibilité

La disponibilité part du principe que les équipements/services peuvent être réparés vers un état fonctionnel après l'occurrence d'une panne. À ce titre, elle représente un indicateur de performance plus réaliste d'un système tel que celui du stockage infonuagique. En somme, maintenir la disponibilité d'un équipement se résume à deux aspects : Le premier consiste à s'assurer que l'équipement ne tombe pas en

panne aussi longtemps que possible. Le deuxième étant de réparer l'équipement aussi rapidement que possible.

Une approche habituellement utilisée pour modéliser la disponibilité est le processus de renouvellement (*renewal process*) (Elsayed, 1996). En résumé, un processus de renouvellement a pour but de décrire la fréquence d'occurrence d'un phénomène (par ex. une panne) du système et se décrit comme étant une alternance de séquences d'états (par ex. état de panne et état de fonctionnement) telle qu'illustrée par la figure 2.2.

Formellement, un processus de renouvellement est défini comme étant une séquence illimitée de variables aléatoires positives, indépendantes et identiquement distribuées dans lequel on considère que le système est réparé à neuf (renouvelé) entre chaque séquence.

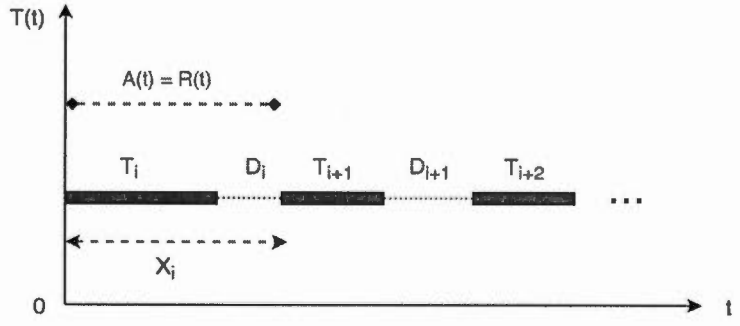

Figure 2.2 Un processus de renouvellement.

Dans le cadre de la disponibilité, nous considérons T_i comme étant la i^e période de fonctionnement et D_i, la i^e période de panne où s'effectue en parallèle des opérations de réparation. Nous avons pour chaque période $i = \{1, 2..., t\}$, une séquence de variables aléatoires $X_i = T_i + D_i$ que nous illustrons à la figure 2.2.

À présent, considérons T_i avec $i = \{1, 2, ..., t\}$, une variable aléatoire positive, indé-

pendante et identiquement distribuée avec une fonction de répartition cumulative $W(t)$ et une densité de probabilité $w(t)$. De façon similaire, nous définissons pour $D_i \mid i = \{1, 2, ..., t\}$, une variable aléatoire positive, indépendante et identiquement distribuée avec une fonction de répartition cumulative $G(t)$ et une densité de probabilité $g(t)$. Par transitivité, X_i avec $i = \{1, 2, ..., t\}$ est également positive, indépendante et identiquement distribuée. X_i avec $i = \{1, 2, ..., t\}$ étant la somme de deux variables aléatoires indépendantes, nous avons par le théorème de convolution, $f(t)$ la densité de probabilité associé à X_i comme étant la convolution entre w et g telle que :

$$\hat{f}(s) = \hat{w}(s)\ \hat{g}(s) \tag{2.17}$$

Il est alors possible de catégoriser l'état de fonctionnement d'un équipement selon deux cas de figures (Elsayed, 1996) :

— Pour le premier cas, l'équipement n'est jamais tombé en panne dans l'intervalle $[0, t]$, correspondant alors à la définition de la fiabilité $R(t)$.

— Le second cas correspond à celui où l'équipement est en état de fonctionnement de nouveau après avoir été réparé à un temps $x \mid 0 < x < t$. La disponibilité équivaut alors à la probabilité $\int_0^t R(t-x)\ m(x)\ dx$.

Combinant ces deux probabilités, la disponibilité est alors définie par la relation suivante (Elsayed, 1996) :

$$A(t) = R(t) + \int_0^t R(t-x)\ m(x)\ dx \tag{2.18}$$

avec $m(x)$ représentant la fonction densité de probabilité du processus de renouvellement du système.

Cette équation se résout en appliquant la transformée de Laplace (Elsayed, 1996) :

$$\hat{A}(s) = \hat{R}(s)\ [1 + \hat{m}(s)] \tag{2.19}$$

$$\hat{A}(s) = \frac{\hat{R}(s)}{1 - \hat{w}(s)\hat{g}(s)} \tag{2.20}$$

avec

$$\hat{m}(s) = \frac{\hat{w}(s)\hat{g}(s)}{1 - \hat{w}(s)\hat{g}(s)} \tag{2.21}$$

En rappelant que $W(t)$ représente la fonction de répartition cumulative dénotant les périodes d'états de fonctionnement du système, on a alors par définition :

$$R(t) = 1 - W(t) \tag{2.22}$$

Par application de la transformée de Laplace, on obtient alors :

$$\hat{R}(s) = \frac{1 - \hat{w}(s)}{s} \tag{2.23}$$

En réécrivant l'équation 2.20 en tenant compte de l'équation 2.23, on obtient :

$$\hat{A}(s) = \frac{1 - \hat{w}(s)}{s[1 - \hat{w}(s)\hat{g}(s)]} \tag{2.24}$$

Nous aurions obtenu la disponibilité instantanée $A(t)$ en effectuant une transformée de Laplace inverse. Cependant, une expression analytique de la transformée de Laplace inverse est difficile à obtenir. C'est pour cette raison que l'on se tourne souvent vers des solutions numériques ou des méthodes d'approximations avec des conditions particulières pour déterminer la disponibilité instantanée $A(t)$.

La disponibilité instantanée se définit comme étant la probabilité qu'un système soit opérationnel à un instant t ponctuel. D'un autre côté, la disponibilité asymptotique est définie comme étant la probabilité qu'un système soit disponible en considérant une période de temps très large (Elsayed, 1996). L'avantage est que l'on peut aisément obtenir une solution analytique et par conséquent, est fréquemment utilisée dans l'analyse de la disponibilité de nombreux systèmes.

En effet, en partant de l'équation 2.24, nous obtenons la disponibilité asymptotique A_s en posant :

$$A_s = \lim_{t \to \infty} A(t) \tag{2.25}$$

En transposant dans le domaine laplacien et en appliquant le théorème de la valeur finale, nous avons :

$$A_s = \lim_{s \to 0} s\hat{A}(s) \tag{2.26}$$

Selon la définition du transformée de Laplace, nous avons :

$$\hat{w}(s) = \int_0^\infty e^{-st}\, w(t)\, dt \tag{2.27}$$

En considérant s très petit (en d'autres termes t très grand), alors nous avons $e^{-st} \cong 1 - st$ telle que :

$$\hat{w}(s) = \int_0^\infty (1 - st)\, w(t)\, dt \tag{2.28}$$

Si nous considérons une distribution exponentielle pour w(t) and g(t), nous dérivons alors :

$$\hat{w}(s) = 1 - \frac{s}{\lambda} \tag{2.29}$$

où λ représente le taux de pannes du système. De façon similaire, nous avons :

$$\hat{g}(s) = 1 - \frac{s}{\mu} \tag{2.30}$$

avec μ, le taux de réparations du système.

Dès lors, nous pouvons reformuler la disponibilité asymptotique comme suit :

$$A_s = \frac{\frac{1}{\lambda}}{\frac{1}{\lambda} + \frac{1}{\mu}} \tag{2.31}$$

En dénotant $\frac{1}{\lambda}$ par le temps moyen de pannes MTTF (*mean time to failure*) et $\frac{1}{\mu}$ par le temps moyen de réparations MTTR (*mean time to repair*), nous retrouvons la formulation usuelle de la disponibilité asymptotique :

$$A_s = \frac{MTTF}{MTTF + MTTR} \tag{2.32}$$

En somme, la disponibilité asymptotique décrite ici s'appuie sur l'hypothèse selon laquelle le taux de pannes durant le cycle de vie du système est constant. En pratique, cette hypothèse suscite beaucoup de critiques et a été largement réfutée à

travers des études récentes (Schroeder et Gibson, 2007; Schroeder *et al.*, 2010; Pinheiro *et al.*, 2007; Elerath, 2000; Elerath et Shah, 2004). En réalité, le taux de pannes des équipements pertinents du système de stockage infonuagique exhibe une variation au fil du temps selon les références (Pinheiro *et al.*, 2007; Schroeder et Gibson, 2007). Par conséquent, supposer une disponibilité asymptotique simplifie excessivement l'analyse de la disponibilité en sous-estimant la disponibilité réelle face aux variations de taux de pannes au fil du temps (Sun et Han, 2001). Il devient ainsi critique, dans le cadre d'un système de stockage infonuagique, de pouvoir déterminer la disponibilité instantanée pour assurer une satisfaction continue des exigences SLA.

Dans le cadre de ce travail, nous calculons la disponibilité instantanée en nous basant sur les travaux de (Sun et Han, 2001) qui fournit une approximation de la disponibilité instantanée tenant compte de la variation des taux de pannes. En effet, ces auteurs partent du principe selon lequel une variation notable du taux de pannes n'est observable que sur une période de temps assez large (par ex. annuel ou mensuel). À ce titre, on peut raisonnablement supposer, pour des périodes de temps moins large (par ex. quelques semaines ou quelques mois), que le taux de pannes varie peu, et par conséquent peut être considéré comme constant.

Ainsi, ils considèrent un système avec un taux de pannes en escalier où le taux de pannes λ transite à un temps h d'un taux de pannes λ_0 à λ_1.

$$\lambda = \begin{cases} \lambda_0 & t \leq h \\ \lambda_1 & t > h \end{cases} \tag{2.33}$$

En combinant les équations 2.11 et 2.12, on déduit la fonction densité de probabilité $f(t)$ comme suit :

$$f(t) = \begin{cases} \lambda_0 \cdot e^{-\lambda_0 t} & t \leq h \\ \lambda_1 \cdot e^{-\lambda_0 t} \cdot e^{-\lambda_1 (t-h)} & t > h \end{cases} \tag{2.34}$$

Par généralisation, nous reformulons les équations 2.33 et 2.34 telle que :

$$\lambda = \begin{cases} \lambda_0 & t \leq h_1 \\ \lambda_{i-1} & h_{i-1} < t \leq h_i, \quad i > 1 \end{cases} \tag{2.35}$$

$$f(t) = \begin{cases} \lambda_0 \cdot e^{-\lambda_0 t} & t \leq h_1 \\ \lambda_{i-1} \cdot \alpha_{i-1} \cdot e^{-\lambda_{i-1} \cdot (t-h_{i-1})} & h_{i-1} < t \leq h_i, \quad i > 1 \end{cases} \tag{2.36}$$

avec :

$$\begin{cases} \alpha_1 = e^{-\lambda_0 h_1} & t \leq h_1 \\ \alpha_i = \alpha_{i-1} \cdot e^{-\lambda_{i-1} \cdot (h_i - h_{i-1})} & h_{i-1} < t \leq h_i, \quad i > 1 \end{cases} \tag{2.37}$$

En outre, on considère également que le système peut être réparé sous un temps constant. Dans le cadre d'un infrastructure de stockage infonuagique, cette hypothèse est raisonnable d'autant plus que selon (Schroeder et Gibson, 2007), le temps moyen nécessaire pour la réparation de tout type de pannes, peut en pratique être estimé à 6 heures. Ainsi, en considérant un taux de pannes et un taux de réparations constant pour un intervalle de temps suffisamment petit (par ex. jours, mois), (Sun et Han, 2001) modélisent la disponibilité en utilisant une chaîne de Markov non-homogène.

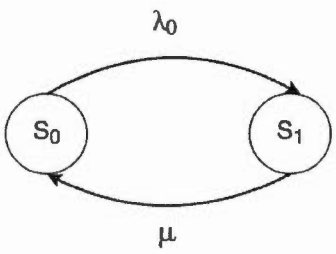

Figure 2.3 Illustration du modèle de Markov d'un système réparable.

Pour $t \leq h_1$, considérons un système de Markov avec un taux de pannes λ_0 et un taux de réparations μ. On dénote alors deux états, mutuellement exclusifs, pour le système : un état de fonctionnement S_0 et un état de panne S_1.

Les états de transitions du système à l'instant $t + \Delta t$ sont alors définis comme suit :

$$P_0\ (t + \Delta t) = [1 - \lambda_0\ \Delta t]\ \cdot\ P_0(t)\ +\ \mu\ \cdot\ \Delta t\ \cdot\ P_1(t) \tag{2.38}$$

$$P_1\ (t + \Delta t) = [1 - \mu\ \Delta t]\ \cdot\ P_1(t)\ +\ \lambda_0\ \cdot\ \Delta t\ \cdot\ P_0(t) \tag{2.39}$$

Par dérivation, nous avons :

$$\frac{dP_0(t)}{dt} = -\lambda_0\ P_0(t)\ +\ \mu\ P_1(t) \tag{2.40}$$

$$\frac{dP_1(t)}{dt} = -\mu\ P_1(t)\ +\ \lambda_0\ \ P_0(t) \tag{2.41}$$

En supposant que le système soit initialement en état de marche, les équations ci-dessus peuvent être résolues en appliquant la transformée de Laplace :

$$s \cdot P_0(s) - P_0(0) = -\lambda_0\ P_0(s)\ +\ \mu\ P_1(s) \tag{2.42}$$

$$s \cdot P_1(s) = -\mu\ P_1(s)\ +\ \lambda_0\ \ P_0(s) \tag{2.43}$$

À partir de l'équation 2.43, nous dérivons la suivante :

$$P_1(s) = \frac{\lambda_0}{s + \mu}\ P_0(s) \tag{2.44}$$

En combinant les équations 2.42 et 2.43, nous avons :

$$P_0(s) = \frac{s + \mu}{s \cdot (s + \lambda_0 + \mu)} \tag{2.45}$$

Par décomposition en fractions partielles, nous reformulons l'équation 2.45 comme suit :

$$P_0(s) = \frac{\frac{\mu}{\lambda_0 + \mu}}{s} + \frac{\frac{\lambda_0}{\lambda_0 + \mu}}{s + \lambda_0 + \mu} \tag{2.46}$$

En appliquant la transformée de Laplace inverse nous obtenons la disponibilité instantanée pour $t \leq h_1$:

$$P_0(t) = A(t) = \frac{\mu}{\lambda_0 + \mu} + \frac{\lambda_0}{\lambda_0 + \mu} \cdot e^{-(\lambda_0+\mu)\ t} \tag{2.47}$$

De manière analogue, pour un intervalle de temps $h_{i-1} < t \leq h_i \mid i > 1$, nous obtenons la disponibilité telle que :

$$P_0(t) = A(t) = \frac{\mu}{\lambda_i + \mu} + \left[A(h_i) - \frac{\mu}{\lambda_i + \mu} e^{-(\lambda_i+\mu)(t-h_i)}\right] \tag{2.48}$$

En combinant les équations 2.47 et 2.48, nous retrouvons la formulation de la disponibilité instantanée proposée par (Sun et Han, 2001) :

$$A(t) = \begin{cases} \frac{\mu}{\lambda_0+\mu} + \frac{\lambda_0}{\lambda_0+\mu} \cdot e^{-(\lambda_0+\mu)\ t} & t \leq h_1 \\ \\ \frac{\mu}{\lambda_i+\mu} + \left[A(h_i) - \frac{\mu}{\lambda_i+\mu} e^{-(\lambda_i+\mu)(t-h_i)}\right] & h_{i-1} < t \leq h_i,\ i > 1 \end{cases} \tag{2.49}$$

Il est important de noter que l'on peut accroître la précision de l'estimation de la disponibilité instantanée en réduisant les intervalles de temps où les taux de pannes varient (Sun et Han, 2001).

La formulation présentée ci-dessus a l'avantage de prendre en compte la variation des taux de pannes au fil du temps tout en proposant une expression analytique simple de la disponibilité instantanée, en supposant raisonnablement que le taux de pannes et le taux de réparations peuvent être considérés comme étant constant pour un intervalle de temps moins large.

Dans la mesure où la solution que l'on propose s'inspire d'un algorithme génétique (détaillé au chapitre 5), nous calculons de manière fréquente une fonction

d'évaluation afin d'évaluer la qualité d'une solution potentielle. Ainsi, disposer d'une expression simple mais néanmoins suffisamment précise est essentiel pour l'efficacité de notre système.

Évidemment, la formulation met également en exergue l'importance du taux de pannes ainsi que la variation de celui-ci dans la valeur calculée. Ainsi, dans le contexte d'une infrastructure de stockage infonuagique avec des disques durs, récipiendaires des données et hétérogènes du point de vue du taux de pannes, cela a une influence non-négligeable sur la disponibilité des données.

Pour ces raisons, nous exploitons cette formulation dans le cadre de la solution que nous présentons dans ce travail.

Dans le contexte de la réplication des données, garantir la disponibilité des données consiste à dupliquer un équipement en plusieurs instances et à les faire fonctionner en parallèle de sorte qu'une instance puisse prendre le relais dès lors qu'une panne se produit. Ainsi, la disponibilité d'un système en configuration parallèle se détermine en calculant la probabilité qu'au moins une copie du système soit fonctionnel. Pour cela, dénotons par $\bar{A}_s$, la probabilité qu'un système soit indisponible. Nous avons alors :

$$\bar{A}_s = P(\bar{e_1}) \cdot P(\bar{e_2}/\bar{e_1}) \cdot P(\bar{e_3}/\bar{e_2}\bar{e_1}) \cdot \ldots \cdot P(\bar{e}_n/\bar{e}_{n-1}\bar{e}_{n-2}\ldots) \tag{2.50}$$

En d'autres termes, le système n'est indisponible que si tous les équipements copies le sont également. Étant mutuellement exclusive, la disponibilité peut être déduite de l'indisponibilité comme suit :

$$A_s = 1 - \bar{A}_s \tag{2.51}$$

En supposant que les équipements soient indépendants, nous avons alors la disponibilité d'un système composé de plusieurs équipements fonctionnant en mode

parallèle comme suit :

$$A_s = 1 - \left[P(\bar{e}_1) \cdot P(\bar{e}_2) \cdot P(\bar{e}_3) \cdot ... \cdot P(\bar{e}_n)\right] \tag{2.52}$$

$$A_s = 1 - \prod_{i=1}^{n} P(\bar{e}_i) \tag{2.53}$$

$$A_s = 1 - \prod_{i=1}^{n} (1 - P(e_i)) \tag{2.54}$$

$$A_s = 1 - \prod_{i=1}^{n} (1 - A_{e_i}) \tag{2.55}$$

Par ailleurs, il est également important de noter que la disponibilité d'un système est généralement décrite à travers le nombre de "9" obtenus ou par un qualificatif dénotant la criticité du système ainsi que l'étendue de la durée d'indisponibilité envisageable (Bauer et Adams, 2012) que nous présentons dans le tableau 2.1. En effet, au-delà du fonctionnement ou non du système, la disponibilité constitue également une métrique quantifiant l'importance du système par rapport à l'accomplissement d'une tâche. Ainsi, selon (Bauer et Adams, 2012; FAA-HDBK-006A, 2008), on peut distinguer trois niveaux de criticité :

— Usuel (99%) : l'indisponibilité du système a un impact mineur dans l'accomplissement de la tâche considérée.

— Essentiel (99.9%) : l'indisponibilité a un impact significatif quant à l'exécution de la tâche.

— Critique (99.999%) : l'indisponibilité mettrait de façon inacceptable en danger la matérialisation de la tâche considérée.

Quant. de 9 / (%)	**Qualification du système**	**Indispo. à l'année (min.)**	**Indispo. au mois (min.)**	**Signification pratique**
1 / 90	Disponibilité non maitrisée	52 596	4 383	Indisponibilité de 5 semaines par an
2 / 99	Disponibilité maitrisée	5 259	438.3	Indisponibilité de 4 jours par an
3 / 99.9	Disponibilité bien maitrisée	525.9	43.83	Indisponibilité de 9 heures par an
4 / 99.99	Tolérant aux pannes	52.6	4.38	Indisponibilité de 1 heure par an
5 / 99.999	Hautement disponible	5.26	0.44	Indisponibilité de 5 minutes par an
6 / 99.9999	Très hautement disponible	0.53	0.04	Indisponibilité de 30 secondes par an
7 / 99.99999	Ultra disponible	0.05	-	Indisponibilité de 3 secondes par an

Tableau 2.1 Degrés de disponibilité d'un système. (GR-2841-CORE, 1994)

CHAPITRE III

REVUE DE LITTÉRATURE

La gestion des réplicas au sein d'une infrastructure de stockage a suscité beaucoup d'attention dans la littérature. Les travaux effectués en ce sens se sont typiquement focalisés à résoudre divers aspects liés au bon fonctionnement et à la rentabilité économique et énergétique des systèmes de stockage tels que l'amélioration de la disponibilité, de la fiabilité, de la cohérence des données ou encore la minimisation de la consommation d'énergie et de l'espace de stockage utilisée par la réplication.

À travers ce chapitre, nous récapitulons brièvement les contributions antérieures qui sont liées aux aspects pertinents à notre étude. Plus précisément, nous nous intéressons aux travaux qui ont trait à la caractérisation et à la prévision des pannes de disques durs ainsi qu'à la gestion des réplicas de données dans le cadre d'une infrastructure de stockage infonuagique hétérogène.

3.1 Caractérisation et prévision des pannes de disque

Les pannes de disque durs ont été rapportées comme étant l'une des causes majeures d'indisponibilité des données et, au pire des cas, des pertes de données (Pinheiro *et al.*, 2007; Schroeder et Gibson, 2007). En conséquence, la caractérisation des pannes de disques durs a retenu l'attention de nombreux travaux de recherche (Pinheiro *et al.*, 2007; Schroeder et Gibson, 2007; Li *et al.*, 2014; Zhu

et al., 2013). À ce titre, (Schroeder et Gibson, 2007) observent que le taux de pannes des disques durs s'accroît au fil du temps avec un taux de remplacement variant communément entre 2% à 4%, atteignant même pour certains systèmes jusqu'à 13%. De plus, ils observent également que le comportement pré-panne (i.e., variation des taux de pannes) variait au niveau même des disques durs de modèles similaires en fonction de paramètres et des conditions de fonctionnement tels que la température ou encore le volume de trafic.

Allant dans ce sens, (Vishwanath et Nagappan, 2010) confirment à travers leur recherche la place prépondérante des pannes de disques durs au sein d'un infrastructure infonuagique. Ainsi, ils observent que les pannes de disques durs représentaient 78% des pannes à la source des pannes de serveur ou d'interruption de services. En outre, ils remarquent également que les serveurs ayant auparavant subi une panne sont davantage prédisposés à des pannes ultérieures dans un futur proche.

(Ford *et al.*, 2010) analysent en détail une variété de paramètres systèmes pouvant provoquer une indisponibilité de données dans une infrastructure de stockage appartenant à Google. Bien que reconnaissant que les pannes de disques durs peuvent être à l'origine de pertes de données, ils estiment que le plus grand obstacle à la disponibilité sont les *explosions de pannes* (*failure burst*) qu'ils définissent comme étant un intervalle de temps durant lequel se produit quasi simultanément un grand nombre de pannes d'équipements.

Dans leur travaux, (Pinheiro *et al.*, 2007) constatent, après avoir groupé les disques durs selon leur âge, que les taux de pannes annuels variaient en moyenne entre 1.7% et 8.6%. Ils pointent également dans leur étude la forte corrélation qui peut exister entre certains attributs S.M.A.R.T (*Self-Monitoring, Analysis and Reporting Technology*) et les pannes de disques durs. S.M.A.R.T est un méca-

nisme de supervision de divers paramètres afin de diagnostiquer et détecter les défaillances mécaniques des disques durs.

Partant de ce constat, de récentes études (Li *et al.*, 2014; Zhu *et al.*, 2013) se sont focalisées sur des modèles de prédiction de pannes de disques durs sur la base de leurs attributs S.M.A.R.T non seulement capable d'atteindre un taux de prédiction très élevé mais également un niveau de faux positifs très bas. Plus précisément, (Li *et al.*, 2014) présentent une approche utilisant, dans un premier temps, un arbre de classification permettant de prédire la panne d'un disque dur. Dans un second temps, ils s'intéressent à quantifier l'état de santé courant d'un disque dur à travers un modèle d'arbre de régression. À la suite de leur étude, ils obtiennent comme résultats un taux global de précision autour de 95% pour un taux de faux positif, qui dénote la proportion de disques dur incorrectement prédite comme tombant en panne, sous 0.1%. Suivant la même logique, (Zhu *et al.*, 2013) proposent quant à eux un modèle de prédiction basé sur les réseaux neuronaux avec rétro-propagation. Les résultats rapportés tournent également autour de 95% de précision pour la prédiction avec un taux de faux positifs de 0.48%.

3.2 Mécanismes de réplication de données dans l'infonuagique

Au cours des dernières années, les problématiques ayant trait à la réplication de données au sein des systèmes infonuagiques ont suscité un vif intérêt auprès des chercheurs, qui se sont focalisés principalement sur la quantification du nombre minimal de réplicas nécessaire ainsi que leurs emplacements dans le système. Tout cela en fonction de plusieurs aspects tels que la minimisation des coûts, de la consommation d'espace de stockage et du temps de réponse ou encore la maximisation de la disponibilité des données (Rahman *et al.*, 2006; Wei *et al.*, 2010; Abad *et al.*, 2011; Li *et al.*, 2011; Sun *et al.*, 2012; Liao *et al.*, 2012; Li *et al.*, 2012; Lin *et al.*, 2013a).

Ainsi, (Abad *et al.*, 2011) ont proposé DARE, un mécanisme de réplication de données pour des clusters MapReduce qui s'adapte à la popularité des fichiers. Cependant, la méthode proposée exige une consommation d'espace de stockage élevée. De plus, les auteurs ne tiennent pas compte de l'hétérogénéité des noeuds de stockage.

Quant à (Liao *et al.*, 2012), ils ont proposé DRDS, une stratégie dynamique de gestion de réplicas qui vise à réduire la consommation d'espace de stockage ainsi que la maintenance liée en supprimant les réplicas non essentiels du point de vue performance tout en garantissant la satisfaction d'exigences QoS. Toutefois, ce travail ne discute pas de la variabilité de la demande ou encore de la variation en termes de taux de pannes de l'infrastructure de stockage.

(Cidon *et al.*, 2013) proposent la réplication Copyset visant à réduire la fréquence d'occurrence de pertes de données dans les infrastructures de stockage infonuagique lors de pannes globales d'énergie. Ainsi, la réplication Copyset se charge de trouver un compromis quasi optimal entre le nombre de noeuds sur lesquels une donnée est distribuée et la probabilité que cette donnée soit perdue lors d'une panne. Cette approche, cependant, ne s'intéresse pas à l'hétérogénéité des noeuds de stockage auxquels les réplicas sont distribuées.

(Harnik *et al.*, 2009) présentent une méthode de réplication dans une infrastructure de stockage infonuagique visant principalement à réduire la consommation électrique, durant les périodes où la demande en termes d'accès aux données est plus faible, en opérant dans un mode de basse puissance énergétique sans toutefois sacrifier d'autres aspects importants du système tels que la disponibilité et la fiabilité. Ils concluent qu'en général, les algorithmes gloutons de réplication permettaient globalement d'obtenir une bonne économie énergétique en comparaison de méthodes de réplication aléatoires. Les résultats que nous trouvons dans notre

étude concordent avec cette conclusion.

(Sun *et al.*, 2012) proposent dans leur travaux une stratégie de réplication dynamique déclenchant la réplication de données, une fois que la popularité du fichier dépasse un seuil défini. Ils établissent également une relation entre la disponibilité et le nombre de réplicas nécessaire. Cependant, leur modèle de centre de données est constitué de noeuds de stockage homogène et, par conséquent, ne tient pas compte de variation de taux de pannes.

Finalement, (Bonvin *et al.*, 2010), proposent une stratégie de réplication autosuffisante, visant à garantir la disponibilité et dans laquelle ils emploient la notion d'économie virtuelle au sein d'un environnement basé sur la théorie des jeux. Cela consiste notamment à faire en sorte que chaque partition de données agisse comme un optimiseur indépendant. Ainsi, selon les bénéfices engendrés en termes de coûts de stockage et de maintenance, une partition de données peut alors décider de migrer, de se répliquer ou de se supprimer. L'objectif global est alors que chaque partition de données jouisse d'un gain maximal.

CHAPITRE IV

VUE D'ENSEMBLE DE HERON : NOTRE PLATEFORME DE GESTION DE RÉPLICAS

Dans ce chapitre, nous proposons d'abord un aperçu de l'environnement d'étude dans lequel nous déployons notre solution. Ainsi, nous expliquons le principe de fonctionnement ainsi que les interactions entre les différentes modules qui composent notre plateforme de gestion de réplicas chargé de garantir la satisfaction des exigences SLA dans le contexte d'une infrastructure de stockage infonuagique hétérogène.

4.1 Principe de fonctionnement

Sommairement, les systèmes de stockage peuvent être vus comme étant des système dont l'objectif est de stocker (ou récupérer) de l'information codée sous forme de fichiers à destination (ou en provenance) d'une infrastructure de stockage composée d'un grand nombre de serveurs, plus précisément de disques durs. Typiquement, chaque fichier est généralement réparti en de multiples blocs de données — que nous appelons par la suite blocs logiques — à travers un mécanisme appelé entrelacement de disques (*data striping*) dont le principe consiste à fragmenter les données en morceaux de données ensuite repartis sur plusieurs disques tout en permettant un accès parallèle. L'avantage est que cela permet d'améliorer les performances d'écriture/lecture. Typiquement, les disques durs sont logiquement

divisés en blocs — que nous dénommons ci-après blocs physiques. Par souci de simplicité, nous considérons que les tous les blocs physiques et logiques sont de tailles fixes. Dès lors, chaque bloc logique est un-à-un mappé à un bloc physique et le contenu du bloc logique est stocké à l'emplacement du bloc physique. Par la suite, le contenu d'un bloc physique peut être répliqué sur d'autres blocs physiques constituant ainsi un emplacement de réplicas. À partir de là, le contenu de chaque bloc logique est accessible à partir de n'importe quel bloc physique à condition que le disque dur contenant celui-ci soit toujours en état de marche.

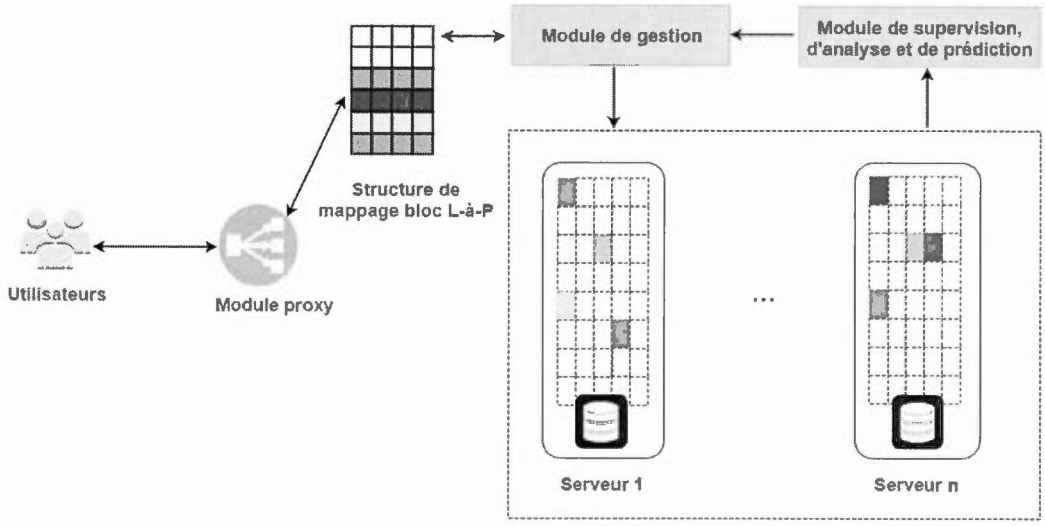

Figure 4.1 Architecture de la plateforme de gestion.

Dans ce contexte, nous présentons notre plateforme de gestion de réplicas, qui est illustrée à la figure 4.1 et qui représente un modèle de gestion de données visant à prendre en compte l'hétérogénéité des équipements de stockage, structuré autour de quatre (4) composants que nous détaillons dans les sections suivantes.

4.2 Structure de mappage entre blocs L-à-P

La structure de mappage permet de mettre en relation les blocs de données et leurs emplacements physiques (c-à-d les disques durs récipiendaires). Elle joue également le rôle d'intermédiaire entre le module proxy et le module de gestion, en étant une couche d'abstraction dans laquelle les opérations de réplication (création, migration et suppression) nécessaires sont exécutées de façon transparente sur des emplacements de réplicas sans que cela n'impacte sur les opérations de lecture/écriture en cours.

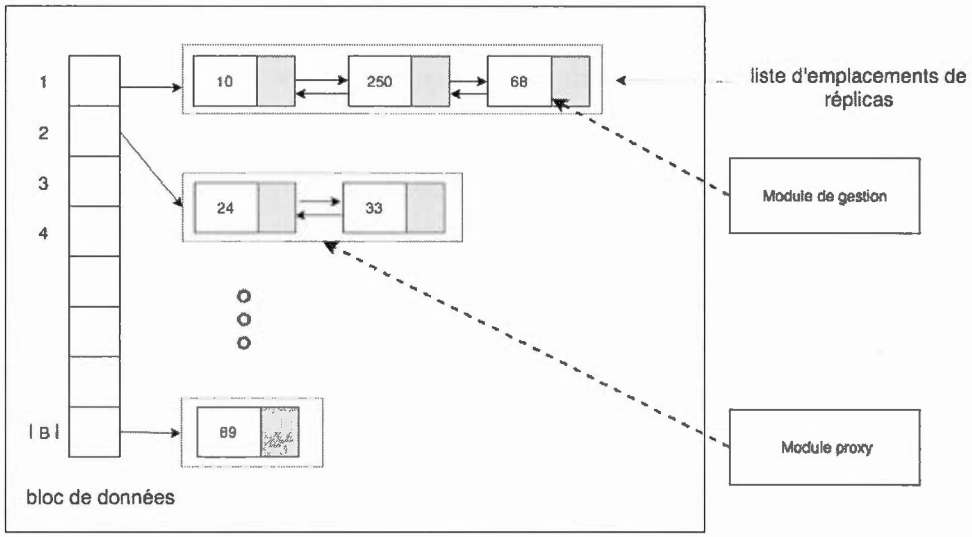

Figure 4.2 La structure de mappage.

Par souci d'efficience et compte tenu du grand nombre de blocs de données stockées au sein de notre infrastructure de stockage, notre structure de mappage est une table de hachage, telle qu'illustrée à la figure 4.2, dans laquelle nous associons chaque bloc de données à une liste d'emplacements de réplicas. Bien évidemment, nous choisissons cette structure de données du fait de sa vitesse en termes d'opérations de recherche, particulièrement face à des conditions strictes SLA en termes

de temps de réponse. En outre, nous notons que la liste d'emplacements de réplicas est une liste doublement chaînée interagissant à la fois avec le module proxy et le module de gestion. Plus précisément, le module de gestion est capable de modifier, d'ajouter et de supprimer des éléments de la liste des emplacements selon les décisions prises pour éviter une violation des exigences SLA. Également, le module de gestion peut aussi, en attente d'une solution, raccourcir temporairement la liste des emplacements lors d'une prédiction de panne d'un disque dur afin d'éviter que des requêtes d'accès y soient routées. D'autre part, le module proxy parcourt la liste d'emplacements pour déterminer les différents disques durs dans lesquels un bloc de données en particulier est stockée.

L'avantage d'une liste chaînée est que les opérations d'insertion et de suppression s'effectuent en temps constant, constituant ainsi un atout essentiel, dans la mesure où un nombre d'opérations potentiellement important peuvent avoir lieu pour garantir la satisfaction des exigences SLA en fonction de l'hétérogénéité de l'infrastructure de stockage.

4.3 Module Proxy

Le module proxy, illustré à la figure 4.3, sert d'interface entre les clients et le système de stockage infonuagique, en ce sens que les clients émettent des requêtes d'accès à leurs données (étape 1) en s'attendant de les recevoir sous les conditions d'accès édictées par SLA. Lorsqu'un fichier est requis par un utilisateur, le proxy vérifie dans la table de mappage (étape 2), décrite précédemment, les emplacements des blocs de données du fichier considéré ainsi que leurs réplicas.

Nous noterons par ailleurs que le module proxy joue également un rôle de répartiteur de charge en distribuant les requêtes d'accès aux différents réplicas en fonction de leur vitesse de traitement en E/S (étape 3). Dès lors, une fois les traitements nécessaires effectués, le module proxy reconstitue le fichier requis (étape 4 et 5)

avant de l'acheminer au client (étape 6).

D'autre part, lorsqu'un fichier est téléversé (*upload*) au niveau du système de stockage infonuagique, le module proxy communique avec le module de gestion (étape 7) afin de s'enquérir sur la manière adéquate de stocker les données dans l'optique de satisfaire les exigences SLA.

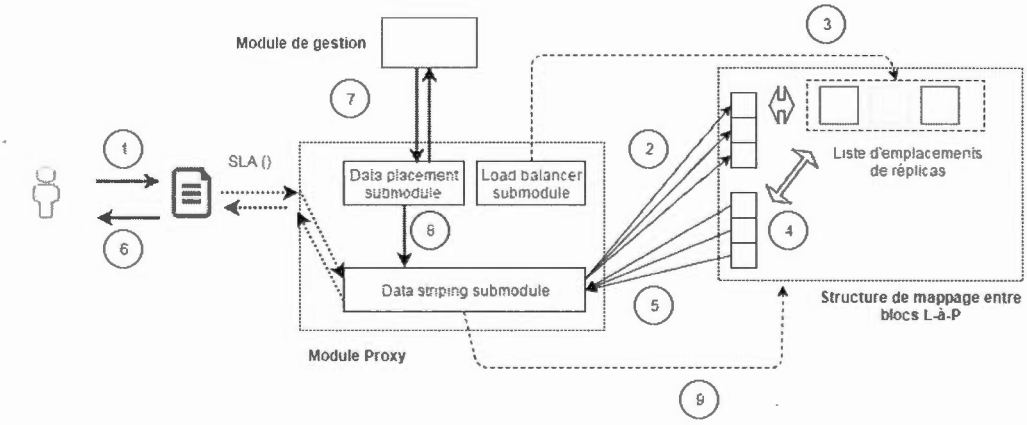

Figure 4.3 Le module proxy.

Une fois que le module de gestion détermine le schéma d'allocation de réplicas approprié pour chaque bloc de données, il copie le contenu des blocs de données et met à jour la structure de mappage entre blocs L-à-P (étape 8 et 9).

4.4 Module de supervision, d'analyse et de prédiction

Une fois qu'une donnée est stockée dans le système de stockage infonuagique, le module de supervision, d'analyse et de prédiction (abrégé module SAP), illustré à la figure 4.4, surveille les requêtes d'accès pour chaque bloc de données (étape 1) et vérifie que les exigences SLA sont bien respectées. Le module SAP effectue également une analyse de ces requêtes (étape 4) afin de déterminer si le module de gestion doit chercher un nouveau schéma d'allocation de réplicas afin que le système puisse continuer à satisfaire les exigences SLA. À cet effet, le module SAP

procède à des opérations de prédiction à partir des données recueillies lors de la supervision.

Tout d'abord, en surveillant le nombre de requêtes par blocs de données, le module SAP tente de détecter les pics futurs de trafic en utilisant un modèle de prévision particulier dénommé ARIMA (*AutoRegressive Integrated Moving Average*) dont nous résumons le principe en annexe A.

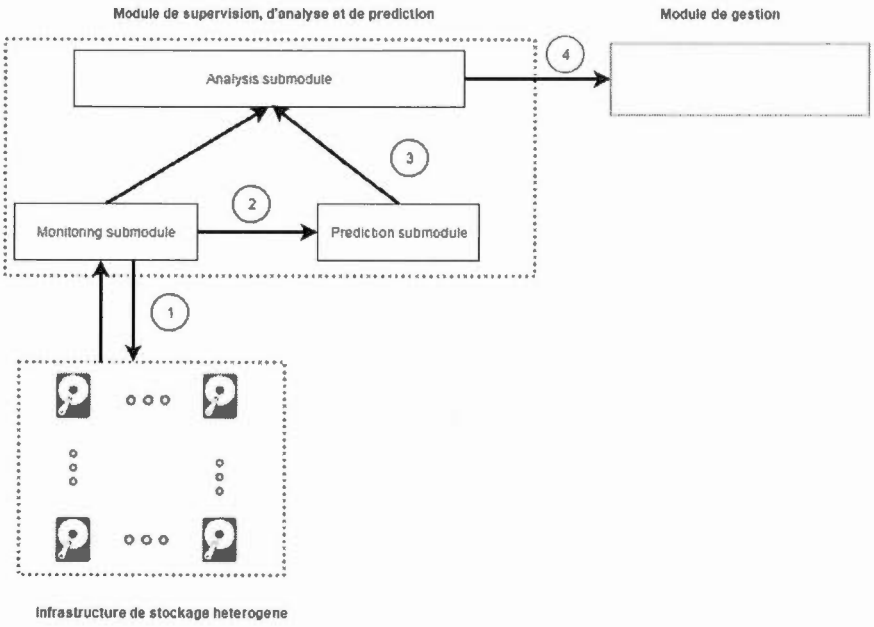

Figure 4.4 Le module SAP.

La raison est que dans un environnement infonuagique, la méthode ARIMA a été démontrée comme étant celle fournissant les meilleurs résultats en termes de précision dans le cadre de la prédiction des requêtes de ressources (Calheiros *et al.*, 2015; Zhao *et al.*, 2012; Zhang *et al.*, 2012; Zhang *et al.*, 2013; Vakilinia *et al.*, 2016). À cet égard, (Calheiros *et al.*, 2015) estiment la précision de prédiction d'ARIMA jusqu'à 91% dans son analyse des traces réels de requêtes en provenance de serveurs web. Dans la même veine, (Zhang *et al.*, 2013) ont analysé, à travers

ARIMA, des traces réelles en provenance des clusters de traitement Google (*Google compute clusters*) pour prédire les taux d'arrivée ainsi que le nombre de tâches. Ils indiquent que l'erreur quadratique moyenne des prédictions (*root square error*) était inférieure à 1%.

Par ailleurs, dans le cadre de ce travail, nous tirons profit de la technique décrite par (Li *et al.*, 2014) qui fait usage d'arbres de classification et d'arbres de régression pour prédire les pannes de disques durs en utilisant les attributs S.M.A.R.T des disques durs. Nous effectuons un résumé de la technique de prévision des disques durs en annexe B.

Toutefois, il est également important de noter qu'au sein de la plateforme que nous proposons, d'autres modèles de prédiction peuvent être facilement utilisés en alternative.

4.5 Module de gestion

Ce module a la responsabilité de déterminer, pour chaque bloc de données au sein du système de stockage infonuagique, le schéma d'allocation de réplicas approprié afin de satisfaire les exigences SLA édictées à travers une heuristique que nous détaillons au prochain chapitre.

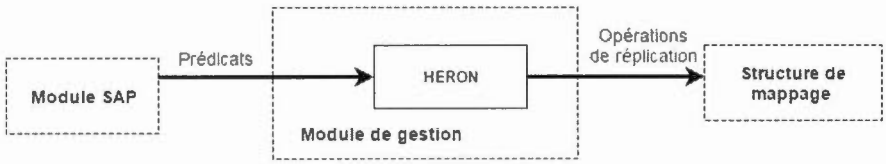

Figure 4.5 Le module de gestion

Tel qu'illustré à la figure 4.5, le module de gestion prend comme paramètres d'entrées les statistiques recueillies ainsi que les prédicats tirés du module SAP pour déterminer un schéma d'allocation stable pour chaque bloc de données du

système.

Une fois qu'une décision est prise concernant les emplacements de réplicas des blocs de données, le module de gestion est aussi responsable de la mise à jour de la structure de mappage entre blocs L-à-P, permettant ainsi lier les blocs logiques et physiques de façon transparente par rapport au module proxy.

Dans ce qui suit, nous décrivons la problématique de la gestion des réplicas dans un système de stockage infonuagique hétérogène en proposant tout d'abord une formulation analytique du problème avant d'analyser sa complexité. En partant de là, nous proposons Heron, une heuristique ayant comme objectif de trouver une solution proche de l'optimal pour chaque bloc de données stocké au sein de l'infrastructure de stockage.

CHAPITRE V

GESTION DES RÉPLICAS DANS UN SYSTÈME DE STOCKAGE INFONUAGIQUE HÉTÉROGÈNE AVEC HERON

En partant de l'environnement d'étude précédemment décrit, nous désirons maintenant formuler mathématiquement le problème de la gestion des données dans une infrastructure de stockage infonuagique hétérogène, de manière à pouvoir analyser les contraintes mais également évaluer la complexité du problème avant de pouvoir proposer une solution proche de l'optimale.

5.1 Formulation du problème

Dans l'optique de simplifier notre analyse, nous supposons un modèle à temps discret $T = \{0, 1, 2, ..., |T|\}$ avec $t \in T$ dénotant une période de temps à laquelle nous prenons une décision de gestion concernant les réplicas (c-à-d. création, migration ou suppression).

En accord avec les études récentes (Vishwanath et Nagappan, 2010; Pinheiro *et al.*, 2007; Schroeder et Gibson, 2007; Schroeder et Gibson, 2010) imputant la majorité des pannes et interruptions de services dans l'infrastructure de stockage infonuagique aux pannes de disques — et plus particulièrement selon (Vishwanath et Nagappan, 2010) qui estiment que 78% des pannes dans le système sont dûs aux disques durs — nous pouvons donc raisonnablement considérer la disponibilité des

données comme étant largement dépendante de la disponibilité en elle-même des disques durs.

Dans ces conditions, nous considérons pour notre modèle, un centre de données constitué d'un ensemble de disques durs que nous notons $H = \{0, 1, 2..., |H|\}$. L'infrastructure de stockage étant hétérogène, supposons qu'il y a M modèles de disques durs différents dans l'infrastructure de stockage. On associe alors chaque disque dur $h \in H$ à un modèle $m \in M$. Pour chaque modèle de disque dur m, on le caractérise à partir d'un ensemble de spécifications fournies par le fabricant : la capacité de stockage $c^m \in \mathbb{N}^*$ et le temps de service l^m, défini comme étant le temps nécessaire afin de traiter une unique requête d'E/S pour le modèle de disque concerné. Enfin, une puissance énergétique à l'état actif et en veille, dénotées respectivement par $p^{m,act} \in \mathbb{R}^+$ et $p^{m,idle} \in \mathbb{R}^+$.

Nous effectuons un mappage entre les disques durs et leurs modèles correspondants à travers une matrice $Q_{|M|\times|H|}$ avec $q^{m,h} = 1$ lorsqu'un disque dur $h \in H$ provient d'un modèle m. Nous pouvons alors dériver le temps de service d'un disque dur comme étant celui de son modèle à travers la relation suivante :

$$s^h = \sum_{m \in M} q^{m,h} \cdot l^m \qquad \forall h \in H, \quad q^{m,h} = \{0, 1\} \tag{5.1}$$

Pour modéliser la variation au fil du temps du taux de pannes des disques durs, nous considérons pour chaque disque $h \in H$ de modèle $m \in M$, un ensemble $\lambda^{m,h} = \{\lambda_1^{m,h}, \lambda_2^{m,h}, ..., \lambda_{|T|}^{m,h}\}$ avec l'élément $\lambda_t^{m,h}$, désignant le taux de pannes observé au temps $t \in T$ pour le disque $h \in H$.

Dans notre analyse, nous supposons que toutes les données appartenant aux utilisateurs sont partitionnées en bloc de données, par la suite mappées au niveau des disques durs. À cet effet, nous dénotons l'ensemble des blocs de données par $B = \{0, 1, 2..., |B|\}$ avec $\bar{b} \in \mathbb{N}^+$ représentant la taille d'un bloc $b \in B$.

Étant donné qu'une donnée stockée n'est disponible que si tous ses fragments le sont également, nous répercutons les exigences SLA des données en provenance des utilisateurs (par ex. des fichiers) sur chacun des blocs de données correspondants. Pour cette raison, nous définissons pour chaque bloc de données $b \in B$, une disponibilité minimale A^b_{thr} et un temps de réponse maximal W^b_{thr}. Cela étant, pour la suite de notre analyse, nous nous focalisons sur la satisfaction des exigences SLA pour les blocs de données.

En outre, nous considérons par souci de simplicité du langage, que tous les blocs de données constituent des réplicas de données. À ce titre, nous définissons une matrice $R_{|B|\times|H|\times|T|}$ qui effectue un mappage entre les emplacements de chaque réplica de bloc de données et les disques durs sur lesquels ils sont placés. L'élément $r^{b,h}_t = 1$ dénote alors le fait qu'un réplica de bloc de données est placé sur un disque dur h. Nous imposons alors deux contraintes pour l'assignation des réplicas :

$$r^{b,h}_t \in \{0, 1\} \tag{5.2}$$

$$\sum_{b \in B} r^{b,h}_t \cdot \bar{b} \leq \sum_{m \in M} q^{m,h} \cdot c^m \quad \forall h \in H, \ \forall t \in T \tag{5.3}$$

La contrainte 5.2 sert à éviter que l'on perde plusieurs — ou pire tous les réplicas — du fait qu'un seul disque soit tombé en panne. De même, la contrainte 5.3 assure que l'on ne dépasse pas la capacité du disque avant que l'on y assigne un réplica de taille $\bar{b}$. Analysons à présent le nombre et le placement des réplicas nécessaires à la satisfaction des exigences relatives à la disponibilité des blocs de données.

5.1.1 Modélisation des contraintes et des coûts de gestion

En considérant que les pannes de disques durs constituent la principale cause d'indisponibilité des données dans un centre de données, nous simplifions notre analyse en supposant que la disponibilité des blocs de données sont identiques

à celui du disque dur sur lequel ils sont stockés. Par là même, nous considérons également que les fluctuations du taux de pannes influençant la disponibilité du disque dur influent également sur la disponibilité des blocs de données.

Ainsi, dans le cadre de ce travail, nous pouvons modéliser la disponibilité des disques durs à travers la méthode présentée au chapitre 2 et qui se base sur les travaux de (Sun et Han, 2001). Étant donné que les pannes de disques durs sont prédites à l'avance, nous pouvons raisonnablement présumer que l'on peut réparer un disque dur sous un temps constant ou à défaut prendre une décision concernant les blocs de données afin de préserver la satisfaction des exigences SLA.

Dès lors, nous calculons pour chaque disque dur $h \in H$ la disponibilité instantanée $A_t^{m,h}$ à l'instant t (voir chapitre 2) :

$$A_t^{m,h} = \begin{cases} \frac{\mu}{\lambda_0^{m,h}+\mu} + \frac{\lambda_0^{m,h}}{\lambda_0^{m,h}+\mu} e^{-(\lambda_0^{m,h}+\mu)t} & \text{si } t \in [0,1] \\ \frac{\mu}{\lambda_i^{m,h}+\mu} + [A_t^{m,h} - \frac{\mu}{\lambda_i^{m,h}+\mu} e^{-(\lambda_i^{m,h}+\mu)(i-t)}] & \text{si } t, i \in [2,3,...,|T|] \end{cases} \tag{5.4}$$

avec $\lambda_t^{m,h}$ et μ dénotant respectivement le taux de pannes et le taux de réparations du disque dur h à l'instant $t \in T$ et l'instant $i = t + 1 \mid i \in T$

Pour garantir la disponibilité des données au sein de l'infrastructure de stockage, nous tirons avantage de la réplication de données, consistant à démultiplier les sources servant les données. Étant donné qu'un bloc de données est disponible, si au moins un réplica est disponible, nous modélisons les réplicas selon une configuration parallèle (précédemment expliqué au chapitre 2). La disponibilité instantanée des blocs de données est alors définie par la formule suivante :

$$A_t^b = 1 - \prod_{h \in H} (1 - (A_t^{m,h} \cdot r_t^{h,b})), \quad \forall b, \forall h, \forall t \tag{5.5}$$

Dans l'optique d'assurer la satisfaction des SLA relatives à la disponibilité, nous

imposons la contrainte suivante :

$$A_t^b \geq A_{thr}^b, \quad \forall b, \forall t \tag{5.6}$$

avec A_{thr}^b l'exigence de disponibilité définie pour le bloc de données b.

Nous nous préoccupons à présent dans notre analyse de satisfaire les exigences en termes de délai de réponse pour les requêtes d'accès à un bloc de données.

Il existe, au sein d'un infrastructure de stockage infonuagique, une hétérogénéité en termes de vitesse d'E/S du fait que les disques durs proviennent de modèles ayant des spécifications différentes, conduisant naturellement à des divergences d'un disque dur à un autre sur les délais de traitement. Cela étant, nous définissons D_t^b, la demande en termes de requêtes d'accès, exprimée en termes de requêtes d'E/S par seconde (*I/O requests per seconds - IOPS*), pour un bloc de données b à un instant t. La demande D_t^b est alors distribuée à travers les différents réplicas du bloc de données b, tout en respectant une contrainte de délai de traitement W_{thr}^b. Nous modélisons à cet effet un disque dur comme étant une file d'attente M/M/1/K dans laquelle patientent les requêtes d'E/S, en attendant d'être servies. Nous déterminons le temps d'attente w_t^h par la relation suivante :

$$w_t^h = \rho_t^h \cdot d_t^{b,h} \left(\frac{1}{1-\rho_t^h} - \frac{(k+1) \cdot (\rho_t^h)^{k+1}}{1-(\rho_t^h)^{k+1}} \right) \leq W_{thr}^b, \quad r_t^{h,b} = 1 \tag{5.7}$$

avec ρ_t^h et $d^{b,h}$ dénotant pour le disque dur h, respectivement l'utilisation de la file d'attente et le taux d'arrivée des requêtes d'E/S. La variable k représente la taille de la file d'attente. À partir de la relation précédente, nous déduisons pour chaque emplacement de réplica d'un bloc b, une partie $d_t^{b,h}$ de la demande globale D_t^b à recevoir tenant bien sur compte des spécifications en termes de vitesse d'E/S du disque dur ainsi que la contrainte de délai de réponse :

$$(d_t^{b,h})^{-1} \geq \frac{\rho_t^h}{W_{thr}^b} \left(\frac{1}{1-\rho_t^h} - \frac{(k+1) \cdot (\rho_t^h)^{k+1}}{1-(\rho_t^h)^{k+1}} \right) \tag{5.8}$$

Nous devons alors nous assurer d'avoir un nombre suffisant de réplicas de données afin de distribuer la demande D_t^b en respectant les contraintes de temps d'accès pour chaque bloc de données :

$$D_t^b = \sum_{h \in H} r_t^{hb} \cdot d_t^{b,h} \quad \forall t, \forall b \tag{5.9}$$

Au-delà du fait que nous devons satisfaire les exigences SLA pour chaque bloc de données dans notre système, nous devons également assurer la stabilité globale du système, en ce sens que les décisions prises (création, migration et suppression de réplicas) pour satisfaire les exigences SLA du bloc de données ne doivent pas induire, au gré des différents paramètres d'hétérogénéité, une fluctuation continue des schémas d'allocation de réplicas. Un schéma d'allocation de réplica de données pour un bloc de données est défini comme étant un vecteur de disques durs sur lesquels sont stockées un réplica du bloc de données.

C'est dans ce cadre que nous traduisons chaque opération de gestion de réplica en un coût monétaire. Ainsi, supposons pour chaque disque dur de modèle m un coût κ^m se rapportant à l'espace de stockage consommé exprimé en termes de \$/MB. Nous pouvons aisément déduire le coût de stockage total induit par un bloc de données b donné par la relation suivante :

$$\kappa_t^b = \sum_{m \in M} \sum_{h \in H} q^{m,h} \cdot \kappa^m \cdot r_t^{b,h} \cdot \bar{b} \quad \forall b, \forall t \tag{5.10}$$

avec c^m représentant la capacité de stockage du disque et $\bar{b}$ la taille du bloc de données b. Nous déduisons alors le coût associé à la création d'un réplica de données :

$$z_t^b = \kappa_{t-1}^b - \kappa_t^b \quad \forall b, \forall t \tag{5.11}$$

De par la relation précédente, il est évident que dans l'optique de réduire le coût global, la dimension (en d'autres termes le nombre de réplicas) du schéma d'allocation de réplica optimal devra être aussi minime que possible.

Outre la consommation d'espace de stockage, nous tenons également à optimiser la consommation énergétique dans notre système. À cet effet, nous considérons une variable binaire $\gamma_t^{b,h} \in \{0,1\}$ indiquant si une opération de lecture/écriture ou de gestion de réplicas est exécutée sur un disque dur h, pour un bloc de données b à un instant t. À l'inverse nous dénotons par $\delta_t^h \in 0,1$ si le disque est au repos ou non à un instant t. Toute opération induit alors un coût énergétique E_t^b (Zedlewski *et al.*, 2003) :

$$E_t^b = (E_t^{b,act} + E_t^{b,idle}) \cdot \phi \quad \forall b, \forall t \tag{5.12}$$

avec ϕ représentant le coût de l'énergie au sein d'un centre de données exprimé en \$/kWh. Les variables $E_t^{b,act}$ et $E_t^{b,idle}$ dénote respectivement l'énergie active et au repos consommées au niveau des disques durs contenant le bloc de données b concerné par l'opération. Elle sont définies par les relations suivantes :

$$E_t^{b,act} = \sum_{m \in M} \sum_{h \in H} q^{m,h} \cdot p^{m,act} \cdot s^h \cdot \gamma_t^{b,h} \cdot \bar{b} \quad \forall b, \forall t \tag{5.13}$$

$$E_t^{b,idle} = \sum_{m \in M} \sum_{h \in H} q^{m,h} \cdot p^{m,idle} \cdot \delta_t^h \quad \forall b, \forall t \tag{5.14}$$

avec $p^{m,act}$ et $p^{m,idle}$ indiquant respectivement la puissance énergétique active et au repos pour le modèle de disque m.

La migration d'un réplica de données constitue un risque à la disponibilité des données puisque que l'on ne peut écarter la possibilité d'une panne durant ce processus. Ainsi, nous infligeons une pénalité ν_t^b à l'opérateur de stockage infonuagique lorsqu'un bloc de données est indisponible durant ce processus. Nous visons, à travers cette pénalité à nous assurer que le meilleur chemin est pris (c-à-d celui présentant le moindre risque de pannes) entre les deux disques impliqués dans le processus de migrations. Ainsi, nous dénotons par $y_{b,t}^{h,u}$ le temps d'arrêt dû à la migration d'un bloc de données b d'un disque dur h à un autre u à un instant t. Nous considérons également un coût monétaire β exprimé en \$/sec. de

sorte que :

$$\nu_t^b = y_{b,t}^{h,u} \cdot \beta \tag{5.15}$$

En combinant les coûts déterminés par les équations 5.11, 5.12 et 5.15, nous formulons le coût total comme suit :

$$C_t^b = z_t^b + E_t^b + \nu_t^b \quad \forall b, \forall t \tag{5.16}$$

Afin de maximiser la stabilité du système ou, en d'autres termes, minimiser le traitement additionnel lié à la gestion de la synchronisation des réplicas, nous devons nous assurer que les schémas d'allocation de réplicas choisis soient assez solides pour, non seulement supporter les fluctuations des taux de pannes des différents disques durs, mais également la variation de la demande pour les blocs de données.

Dans notre travail, nous faisons usage d'un module de supervision, de prédiction et d'analyse, chargé de prédire les pannes de disques durs ainsi que le trafic du centre de données durant un intervalle de temps donné. Cependant, il est important que nous veillions à calibrer adéquatement l'étendue des prédictions dans le temps pour non seulement éviter des coûts inutiles dus à un sur-approvisionnement de ressources mais également pour conserver une meilleure précision des valeurs prédites.

Ainsi, soit $O = \{t, t+1, ..., t+|O|\}$, la portée temporelle des prédictions considérées. Nous nous intéressons dans ce travail, à découvrir les schémas d'allocation de réplicas permettant de minimiser globalement les coûts tout en assurant la satisfaction des exigences SLA pour chaque bloc de données. L'objectif global est donc de minimiser les coûts amortis à travers la fonction objective ci-dessous tout en respectant les contraintes (5.6), (5.8) et (5.9).

$$min \frac{1}{|O|} \sum_{t=0}^{T} C_t^b \quad \forall b \in B \tag{5.17}$$

Dans notre problématique, nous optons d'optimiser les coûts amortis dans la mesure où une décision optimale prise à un instant t peut s'avérer être mauvaise ultérieurement et vice-versa. Ainsi, en incluant les prédictions dans notre fonction objectif, nous remplissons nos objectifs SLA tout en réduisant les coûts au fil du temps.

5.1.2 Analyse de la complexité asymptotique

On pourra aisément constater que la problématique décrite ci-dessus possède un nombre de contraintes et de paramètres exponentiels rendant sa résolution en un temps raisonnable très difficile.

Pour nous convaincre de cela, supposons une version relaxée de la problématique décrite ci-dessus — que nous dénommons $Heron^{relax}$ — et dans laquelle nous éliminons les contraintes 5.2, 5.12 et 5.15. Supposons également, par souci de simplicité, que pour un disque dur h, les taux de pannes $\lambda^{m,h} = \{\lambda_1^{m,h}\}$ et qu'en outre tous les disques durs ont une capacité de stockage constante c ainsi qu'un coût de stockage κ constant.

À travers les conditions édictées ci-dessus, nous avons alors $\mathcal{C}_t^b = z_t^b$. On constate facilement qu'optimiser les coûts de gestion de réplicas revient alors à minimiser le nombre de disques durs utilisés pour stocker les réplicas.

Ainsi, soit n le nombre de réplicas de données nécessaires pour satisfaire les contraintes (5.3, 5.6, 5.8 et 5.9) avec $\bar{b_j} \in \mathbb{N}^+$ dénotant la taille du bloc de données $j \mid j \in N = \{1, 2, ..., n\}$. Soit également $K \in N^*$ représentant le nombre de disques durs utilisés dans l'ensemble des disques durs $H = \{0, 1, 2..., |H|\}$ pour stocker le nombre de réplicas de données n, au sein de notre infrastructure de stockage infonuagique.

$Heron^{relax}$ peut donc être formulé mathématiquement comme suit :

$$min\ K = \sum_{i=1}^{|H|} x_i \tag{5.18}$$

Tel que :

$$\sum_{j=1}^{n} \bar{b_j} \cdot y_{i,j} \leq c \cdot x_i, \qquad \forall i \in H \tag{5.19}$$

$$\sum_{j=1}^{n} y_{i,j} = 1, \qquad \forall j \in N \tag{5.20}$$

$$y_{i,j} \in \{0,1\}, \qquad \forall i \in H, \forall j \in N \tag{5.21}$$

$$x_i \in \{0,1\}, \qquad \forall i \in H \tag{5.22}$$

avec x_i, une variable binaire qui vaut 1 si un disque dur i est utilisé pour stocker un réplica. La variable binaire $y_{i,j} = 1$ dénote le fait qu'un réplica de données j est stocké sur un disque dur i. Ainsi, l'objectif d'$Heron^{relax}$ (5.18) consiste à utiliser la moindre quantité de disques durs pour stocker les réplicas de données tout en respectant les contraintes 5.19 – 5.22. La première contrainte (5.19) permet de s'assurer que l'on ne dépasse pas la capacité de stockage pour chaque disque dur. Les contraintes suivantes 5.20, 5.21 permettent de garantir que chaque réplica de données est assigné à un unique disque dur. La dernière contrainte indique, quant à elle, si un disque dur est utilisé pour stocker un réplica ou non.

Nous pouvons dès lors assimiler $Heron^{relax}$ à une généralisation du problème de *bin packing* (*bin packing problem* - *BPP*) qui a été largement prouvé et classé comme étant un problème NP-difficile (Coffman *et al.*, 1997). En conséquence, nous pouvons également classer $Heron^{relax}$ comme étant NP-difficile.

La problématique précédemment décrite est une extension d'$Heron^{relax}$, en ce sens qu'elle incorpore des contraintes ainsi que des facteurs temporels et d'hétérogénéité en termes de taux pannes pour les disques durs qui ne font qu'ajouter des dimensions de calcul supplémentaires à $Heron^{relax}$. Donc, par transition, nous pouvons considérer la problématique comme étant un problème NP-difficile.

Le BPP consiste à ranger des objets de volumes différents à l'intérieur d'un nombre fini de conteneurs de volume fixe V avec comme objectif d'utiliser le minimum de conteneurs possibles. Étant donné un ensemble de conteneurs $C_1, C_2, ..., C_n$ ayant chacun une capacité fixe $\bar{c}$. Soit alors à ranger n objets de poids $w_1, w_2, ..., w_n$. Une formulation mathématique du BPP est proposée comme suit (Martello et Toth, 1990) :

$$min \sum_{i=1}^{n} y_i \tag{5.23}$$

Tel que :

$$\sum_{j=1}^{n} w_j \cdot x_{i,j} \leq \bar{c} \cdot y_i, \qquad \forall i \in N = \{1, 2, ..., n\} \tag{5.24}$$

$$\sum_{j=1}^{n} x_{i,j} = 1, \qquad \forall j \in N \tag{5.25}$$

$$y_i \in \{0, 1\}, \qquad \forall i \in N \tag{5.26}$$

$$x_{i,j} \in \{0, 1\}, \qquad \forall i \in N, \forall j \in N \tag{5.27}$$

où y_i est égal à 1 si le conteneur C_i est utilisé et $x_{i,j} = 1$ si un objet j est assigné à un conteneur C_i.

Le BPP a été largement étudié dans la littérature. À cet effet, plusieurs contributions ont été faites dans le but de proposer des stratégies d'approximation proche de l'optimal. Ainsi, on peut citer parmi les approches visant à résoudre le BPP, le meilleur ajustement décroissant (*best fit decreasing - BFD*) et le premier ajustement décroissant (*first fit decreasing - FFD*). Le principe de ces deux approches est de trier d'abord les objets par ordre de poids décroissant avant de chercher à caser chaque objet, respectivement dans le meilleur emplacement (celui laissant moins d'espace libre par conteneur) et le premier emplacement disponible. Il a été démontré que l'approche FFD constitue un algorithme borné à au plus 11/9-approché de l'optimal + 1 conteneur (Yue, 1991). Par ailleurs, les

auteurs de (Dósa, 2007) prouvent qu'il est possible d'avoir une borne plus précise de 11/9 - approché de l'optimal + 6/9. D'un autre coté, (Yue et Zhang, 1995) proposent une variante de l'approche FFD, le MFFD dans laquelle une borne de 71/60-approché de l'optimal + 1 conteneur est prouvée.

Cependant, vue la complexité du problème, pour un scénario de grande taille similaire à une infrastructure de stockage infonuagique où il existe de milliers de disques avec des paramètres hétérogènes en plus de facteurs temporels et énergétiques, il est clair que ces techniques ne peuvent pas permettre de trouver une solution optimale dans les temps requis. C'est dans ce cadre que nous proposons, dans la section suivante, une heuristique dont l'objectif est de trouver une solution proche de l'optimale dans un temps raisonnable.

5.2 Solution proposée : HERON

À présent, nous détaillons la conception d'Heron, plus particulièrement l'heuristique sous-jacente, basée sur un algorithme génétique (*Genetic Algorithm - GA*) et qui vise à trouver un schéma d'allocation de réplicas proche de l'optimal pour chaque bloc de données stockés au sein de l'infrastructure de stockage infonuagique.

5.2.1 Description du fonctionnement de l'heuristique

Un GA est une variante d'algorithmes de recherche évolutionnaire, popularisés sous l'impulsion des travaux de John Holland (Holland, 1975), et construit autour des principes de l'évolution biologique naturelle pour résoudre des problèmes d'optimisation complexes. Selon ces principes, chaque individu d'une population est contraint d'améliorer ou encore d'adapter ses propriétés biologiques s'il veut espérer survivre face aux nombreux défis des générations(Mitchell, 1996).

Un des avantages principaux d'un GA provient du fait qu'il permet d'utiliser des informations antérieures pour réorienter de manière efficiente la recherche d'informations vers des régions plus prometteuses du vaste espace de recherche, pour obtenir des solutions quasi-optimales (Ye *et al.*, 2011).

Nous nous intéressons aux GAs pour résoudre notre problème pour plusieurs raisons. Tout d'abord, ils représentent (les GAs) une approche intéressante dans le cas où le problème considéré dispose d'un grand nombre d'optimum locaux. Il est facile de voir que dans notre cas, un schéma d'allocation considéré comme optimal à un instant t, peut devenir problématique quelques instants plus tard au gré des variations des taux de pannes des disques et de la demande. Également, du fait de l'hétérogénéité des équipements, du nombre de paramètres à tenir en compte ainsi que la contrainte de temps à laquelle nous devons trouver une solution alternative pour éviter une violation des exigences SLA, la solution optimale de notre problème consiste à trouver le minimum d'une fonction multi-objective complexe. Ce dernier aspect représente un obstacle pour des méthodes à base de gradients (*gradient based methods*) puisqu'elles nécessitent l'existence d'un gradient de la fonction objective qui n'est pas toujours disponible ou qui est complexe à obtenir, en plus du fait qu'elles considèrent l'ensemble de l'espace de recherche pour trouver une solution globale. Une conséquence de cela est que les méthodes à base de gradients peuvent se retrouver bloquées dans des optimums locaux en plus de ne pas trouver de solutions optimales dans un temps convenable, à l'inverse des GAs qui sont plutôt des solutions se focalisant sur des régions particulières de l'espace de recherche pour converger vers une solution optimale ou suffisamment proche de l'optimale (Mitchell, 1996; Ye *et al.*, 2011).

Également, les GAs sont des méthodes facilement parallélisables, en ce sens que l'exploration se fait simultanément en plusieurs points de l'espace de recherche ce qui permet d'éviter le blocage sur un optimum local à la différence d'algorithmes

classiques partant d'un seul point de l'espace. Par conséquent, nous pouvons aisément réduire le temps nécessaire à la découverte d'une solution optimale, avant le temps limite pour éviter une violation des exigences SLA, en distribuant l'exécution de l'algorithme sur plusieurs machines.

On notera quand même que pour des problèmes de petite envergure, les GAs, étant une méthode stochastique, ne représentent pas souvent la meilleure solution (en termes d'efficacité). Dans ce cas, on devrait plutôt se tourner vers des méthodes exactes/à base de gradients. Toutefois, pour notre problème, on peut considérer que l'hétérogénéité des équipements de stockage ne pose réellement problème qu'au sein d'une infrastructure de grande envergure puisqu'un processus d'homogénéisation serait presque impossible pour des questions économiques et logistiques (Sharma *et al.*, 2011) d'où l'intérêt de notre étude. Ainsi, pour la suite de notre analyse, nous ne considérons qu'une infrastructure de stockage infonuagique de grande envergure (de l'ordre de plusieurs milliers de disques durs).

Le point le plus important pour nous est que les GAs intègrent de façon inhérente l'aspect hétérogène de l'espace de recherche. En effet, en théorie, la diversité de l'espace de recherche permet aux GAs de converger plus rapidement vers la solution optimale à travers l'opérateur de croisement qui tire également profit des informations accumulées au fil des générations (Mitchell, 1996; Ye *et al.*, 2011).

Rappelons brièvement qu'en biologie, les organismes vivants se composent de cellules dont les noyaux comportent des chromosomes formant des chaînes d'ADN. Sur chacun des chromosomes, il existe une suite de nucléotides constituant une chaîne, le gène, dont l'utilité est de coder des fonctionnalités spécifiques de l'organisme comme la couleur des cheveux, l'intelligence, la taille, etc. Cela étant, selon la théorie néo-darwinienne de l'évolution, les gènes d'une population donnée destinées à subsister au fil du temps sont ceux qui sont les plus adaptés aux besoins de

l'espèce par rapport à son environnement. Ce processus d'adaptation se concrétise à travers une phase de reproduction inter-individus dans laquelle une génération d'individus moins adaptés sont éliminés afin d'améliorer la capacité d'adaptation de générations futures (Mitchell, 1996).

De façon analogique, les algorithmes génétiques codifient les paramètres du problème d'optimisation comme étant une combinaison de gènes formant des chromosomes. On associe dès lors les solutions potentielles du problème d'optimisation à un chromosome qui dénote un point de l'espace de recherche. Par l'intermédiaire d'un fonction d'évaluation, ou encore fonction *fitness*, ce chromosome est évalué afin de quantifier sa proximité par rapport à la solution optimale (Cormier, ; Mitchell1996). Ainsi, par élimination régulière de chromosomes non adaptés à chaque génération et en raffinant les qualités des chromosomes subsistants par un couplage de chromosomes parents ayant les meilleurs atouts génétiques, il peut être prouvé que l'on converge progressivement vers une solution optimale ou quasi optimale (Cerf, 1994).

Dans le cadre de notre étude, après avoir caractérisé efficacement l'hétérogénéité l'infrastructure de stockage, nous tirons profit à la fois des informations antérieures ainsi que des prédictions obtenues concernant la demande et les pannes de disques durs pour satisfaire les exigences SLA tout en minimisant les coûts de gestion.

La première étape d'exécution d'un algorithme génétique consiste à transposer adéquatement le problème d'optimisation ainsi que ses paramètres en des termes génétiques (i.e. chromosome, gène, etc.) étant donné que l'efficacité de l'algorithme génétique en dépend fortement (Tollari, 2003). Ainsi, chaque solution potentielle du problème d'optimisation sera encodé sous la forme d'un chromosome avec des gènes (i.e paramètres de la solution) sur lequel on applique un processus évolutif. L'avantage de ce processus de codage étant qu'il offre une abstraction du problème

d'optimisation dans la mesure où il permet d'appliquer un processus plus ou moins générique à des problèmes spécifiques.

Pour notre étude, nous associons à chaque bloc de données, un schéma d'allocation de réplicas représentant le nombre et l'emplacement des réplicas du bloc de données nécessaires à la satisfaction des exigences SLA. Ainsi, dès lors que nous détectons une possibilité que ces conditions ne pourraient plus être satisfaites dans un futur proche (c-à-d. prévision de pannes de disques, prévision de pic ou tombée significative de la demande, etc.), le module de gestion se charge alors de trouver un schéma d'allocation de réplicas alternative pour le bloc de données concerné, déclenchant de ce fait, l'exécution de l'heuristique présenté ici.

Notre modèle de codage consiste à associer chaque schéma candidat d'allocation de réplicas à un chromosome, représentant ainsi une solution potentielle du problème. Un schéma candidat d'allocation de réplicas a comme gènes, des disques hétérogènes tant du point de vue de la variation du taux de pannes au fil du temps mais également en termes de capacités de stockage, de traitement et de vitesse d'E/S. Nous tirons alors profit du codage par valeur où chaque gène est représenté par l'identifiant unique du disque contenant un réplica du bloc de données.

Le codage par valeur réelle constitue souvent la forme de représentation du chromosome la plus convenable et réaliste pour la résolution de problèmes d'optimisation dans la mesure où il permet de coder le chromosome en utilisant une séquence de valeurs liées au problème. Ainsi, on pourra utiliser des nombres réels (Montana et Davis, 1989; Schulze-Kremer, 1994), des chaînes de caractères(Kitano, 1990), etc. (Mitchell, 1996).

Ce type de codage s'avère utile pour des problèmes spécifiques dont il est nécessaire de modifier les opérateurs de croisement et de mutation afin de tenir compte des contraintes particulières du problème considéré. Il est à noter que dans ses travaux,

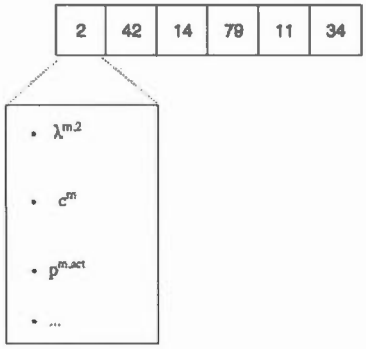

Figure 5.1 Représentation d'un chromosome dans Heron.

John Holland (Holland, 1975), afin d'étayer la pertinence et les performances du codage binaire, semble également indiquer que le codage par valeur serait plus enclin à des performances moindres (Mitchell, 1996). Cette hypothèse a cependant été contestée par des études empiriques (Antonisse, 1989; Janikow et Michalewicz, 1991; Wright *et al.*, 1991) démontrant notamment que, dans certains cas, le codage par valeur du chromosome permet des performances meilleures (Mitchell, 1996).

À la figure 5.1, nous illustrons un chromosome codé dans Heron représentant un schéma d'allocation de réplicas alternatif pour un bloc de données dont on détecte une possibilité de violation des exigences SLA. Chaque gène du chromosome est codé par l'identifiant de disque dur, ce qui nous permet d'obtenir les différents paramètres (modèle, consommation d'énergie) du disque dur.

5.2.2 Mécanisme de réduction de l'espace de recherche

Les algorithmes génétiques constituent une classe d'algorithmes opérant globalement sur une population d'individus. Ainsi, une fois l'étape de représentation du chromosome achevée, la phase suivante consiste à générer une population initiale constituée de chromosomes, représentant chacun une solution potentielle du problème. Il est important de noter que l'efficacité de l'algorithme génétique dépend

également de la taille de la population considérée, puisque l'on part du principe selon lequel la probabilité de trouver les meilleurs solutions s'accroît en accord avec celle-ci (la taille) ainsi que la diversité des individus composant la population initiale. En même temps, une population de grande taille augmente la complexité temporelle de l'algorithme, ralentissant ainsi la convergence.

Étant donné la quantité de disques durs pouvant constituer une infrastructure de stockage infonuagique (de l'ordre de centaines de milliers), auquel s'ajoute pour chaque disque dur divers paramètres hétérogènes, il est essentiel qu'Heron soit capable de prendre une décision concernant le schéma d'allocation de réplicas alternatif à appliquer pour le bloc de données concerné tout en continuant de garantir les exigences SLA.

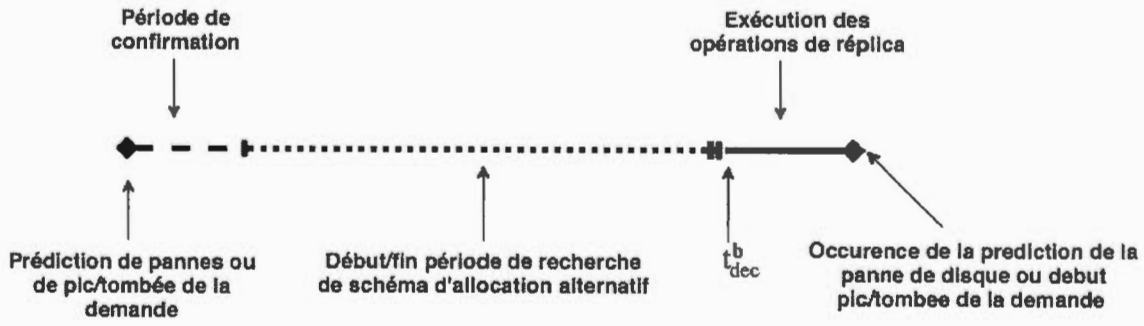

Figure 5.2 Étapes avant l'exécution des opérations de réplication.

Tel que nous le représentons à la figure 5.2, nous définissons pour chaque bloc de données un temps de décision t^b_{dec} auquel un schéma d'allocation alternatif doit être trouvé et à partir duquel les opérations de réplication sont exécutées (création, migration et suppression de réplicas). On notera également qu'avant toute prise de décision, il existe pour chaque bloc de données une période de confirmation dans laquelle nous nous assurons que la prévision est exacte. Dans le cas des disques durs cela se justifie étant donné, qu'il n'est pas rare d'observer des pannes

temporaires dont la durée ne dépasse pas, en général, plus de quinze (15) minutes (Ford *et al.*, 2010).

Dans la mesure où il y a deux aspects principaux (pannes de disques ou pic/tombée notable de la demande) nécessitant que l'on cherche un schéma d'allocation alternatif, le temps de décision t^b_{dec} est contraint par la relation suivante :

$$t^b_{dec} = min(t_f, t_w) \quad \forall b \in B \tag{5.28}$$

où nous dénotons par t_f et t_w, les temps où nous prévoyons respectivement une panne de disque et un pic/tombée de la demande concernant le bloc de données concerné.

Étant donné que la demande peut varier de manière assez fréquente et afin de garder une certaine stabilité du système, nous devons également nous assurer de ne chercher un schéma d'allocation alternatif que lorsque la différence entre la demande actuelle et celle prédite est notable. Nous avons donc, au sein d'Heron, la condition suivante :

$$|D^b_{t_{cur}} - D^b_{t_w}| \leq D^b_{thr} \quad \forall b \in B \tag{5.29}$$

avec $D^b_{t_{cur}}$, $D^b_{t_w}$ et D^b_{thr} dénotant respectivement la demande reçue présentement, celle prédite et la différence seuil.

C'est dans ce cadre que nous définissons un mécanisme de réduction de l'espace de recherche dont l'objectif est d'améliorer l'efficacité et la robustesse d'Heron, en biaisant la recherche de solutions vers des régions plus prometteuses de l'espace de recherche, en tenant compte de la sévérité des exigences SLA. Une fois l'espace de recherche réduit, nous laissons à un algorithme génétique (détaillé plus bas), le soin de l'exploration des solutions potentielles.

Une première étape dans le mécanisme de réduction de l'espace de recherche consiste à évaluer globalement le niveau d'hétérogénéité du système de stockage

infonuagique afin d'identifier les paramètres de disques sur lesquels se focaliseront la recherche.

Nous introduisons dès lors la notion d'ensemble de disques "premium". Dans la mesure où la disponibilité représente la préoccupation majeure d'Heron, nous définissons l'ensemble des disques durs premium comme étant celui regroupant les disques durs dont le taux de pannes évalué est en dessous d'un taux de pannes seuil $f_t^{b,thr}$. La valeur de cette dernière est obtenue comme étant le centroïde minimum d'un partitionnement en k-moyennes (Hartigan et Wong, 1979) — avec $k = 3$ — appliqué sur l'ensemble des taux de pannes des disques durs de l'infrastructure de stockage infonuagique.

Nous considérons également cinq (5) paramètres de disques principaux dans le cadre de notre mécanisme de réduction de l'espace :

- la puissance énergétique active $p^{m,act}$ servant à indiquer la consommation énergétique potentielle, plus particulièrement si le disque dur se devait de contenir des blocs de données qui seront en forte demande ;
- l'écart-type σ^h des taux de pannes typiques du modèle de disque comme indication du degré de fluctuation au fil du temps ;
- la moyenne des taux de pannes du disque $\overline{\lambda}^{m,h}$;
- le temps de service s^h du disque qui nous servira à estimer la vitesse d'E/S du disque ;
- l'utilisation moyenne du disque π^h en vertu de la corrélation qui existe entre l'utilisation du disque et le temps de réponse.

Ainsi, le mécanisme de réduction de l'espace de recherche, en fonction de la proportion de disques premium par rapport à la population entière de disques durs au sein de l'infrastructure de stockage infonuagique, se focalise sur des disques ayant des paramètres particuliers qui ont tendance à être plus compliqués à optimiser au

niveau du système de stockage infonuagique. Par exemple, pour une infrastructure dont la proportion de disques durs premium est très large, nous pensons qu'il est plus judicieux de focaliser la recherche de solutions en optimisant plutôt le nombre et l'emplacement de réplicas nécessaires pour satisfaire les exigences SLA ayant trait au temps de réponse des requêtes d'accès.

Ainsi, nous définissons dans Heron deux notions : un ensemble de paramètres de disques durs primaires et un ensemble de paramètres de disques durs secondaires qui dénotent respectivement les paramètres plus ou moins cruciaux à la satisfaction des exigences SLA du bloc de données concerné. Évidemment, les paramètres retenus dépendent de la sévérité des exigences SLA du bloc de données.

L'objectif dans ce cas est d'associer un profil de régions, au sein de l'espace de recherche en fonction de la sévérité des exigences SLA ainsi que la proportion des disques premium, dans l'optique de réduire non seulement la dimension de l'espace de recherche mais également afin de contrôler l'hétérogénéité des disques durs. Pour cela, plusieurs méthodes sont envisageables : apprentissage supervisé, apprentissage non supervisé, réseaux de neurones, arbres de décision, etc.

Pour Heron, nous faisons plutôt usage d'arbres de décision pour déterminer les paramètres de disques primaires et secondaires. Pour un centre de données ayant une très forte proportion de disques durs premium (par ex. plus de 90%) et pour des blocs de données ayant des exigences en termes de disponibilité élevées (par ex. plus de 99.99%), nous illustrons à la figure 5.3 un exemple d'arbre de décisions simplifié pour déterminer les paramètres de disques primaires et secondaires. De façon plus détaillée, nous considérons que dans ce type d'environnement, étant donné que la plupart des disques durs ont un faible taux de pannes, nous devons focaliser l'effort de recherche d'emplacements de réplicas principalement sur les paramètres d'écart type σ^h et de temps de service s^h comme paramètres primaires.

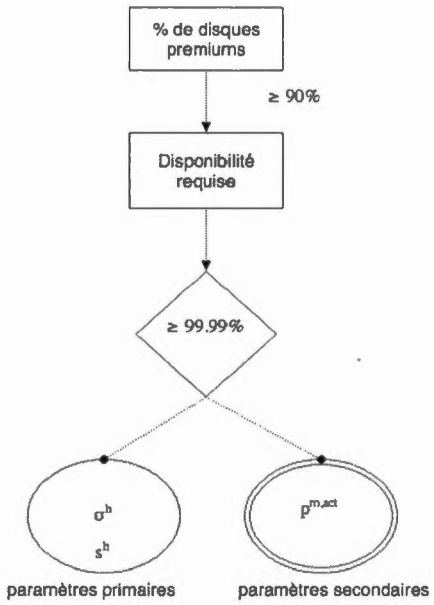

Figure 5.3 Arbre de décision simplifié.

On peut alors considérer la puissance énergétique active du disque, $p^{m,act}$, comme étant un paramètre secondaire.

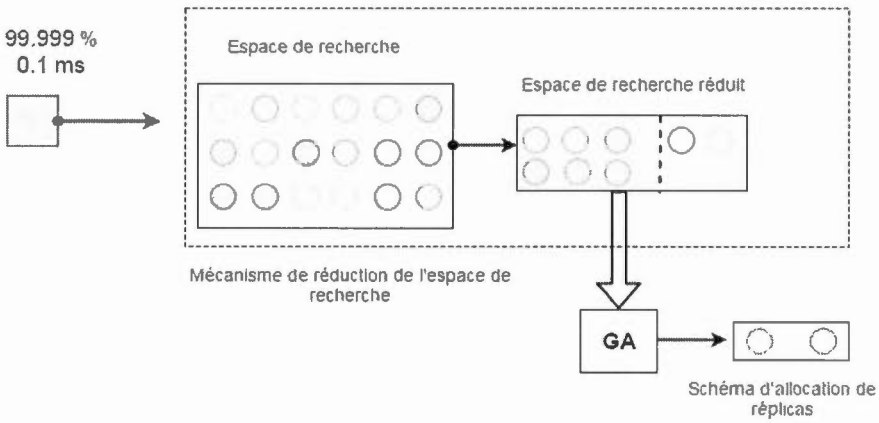

Figure 5.4 Mécanisme de réduction de l'espace de recherche.

Dès lors que les paramètres primaires et secondaires sont définis, nous construisons

un espace de recherche réduit constitué d'un groupe principal et d'un groupe de diversité contenant des disques durs regroupés respectivement en fonction des paramètres primaires et secondaires avec des dimensions variables selon la sévérité des exigences SLA. Nous illustrons l'espace de recherche réduit à la figure 5.4. Pour chaque groupe, nous désignons un disque dur de référence par rapport aux paramètres considérés autour duquel nous regroupons les disques durs similaires. Ainsi, nous appliquons une fonction de similarité définie comme suit :

$$sim(h_i, h_j) = \frac{1}{1 + \sqrt{\sum_{k \in P} (h_i^k - h_j^k)^2}} \tag{5.30}$$

avec $k \in P$ représentant les paramètres de disques et h_i le disque de référence considéré. Les disques ayant le meilleur score sont sélectionnés en s'assurant bien sûr qu'ils ont une capacité de stockage suffisante pour le bloc de données. Lorsque des disques durs ont des scores similaires, l'utilisation moyenne π^h est utilisée pour les départager.

Une fois la population initiale établie, une évaluation de la qualité de chaque individu de la population est faite à travers une fonction d'évaluation (encore fonction *fitness* ou encore fonction objectif). Cette fonction détermine la pertinence d'une solution potentielle du problème, permettant de ce fait de promouvoir ou refuser la participation d'un individu jugé non adapté à la phase de reproduction de la population (principe du *survival of the fittest* de Darwin). Ainsi, elle (la fonction d'évaluation) permet une élimination graduelle des individus de qualité moindre et la subsistance des individus ayant des caractéristiques similaires à la solution optimale du problème d'optimisation.

Nous présentons le fonctionnement du mécanisme de réduction de l'espace à travers l'algorithme 1.

Algorithme 1 Mécanisme de réduction de l'espace de recherche

Entrées : Population entière de disques durs $H = \{1, 2, ..., |H|\}$, A^b_{thr}, $D^b = \{D^b_{t-i}, ..., D^b_{t-1}, D^b_t, D^b_{t+1}, ..., D^b_{t+o}\}$, arbre de décision P

Sortie : Population réduite de disques durs $I = \{1, 2, ..., |I|\}$ avec $|I| < |H|$

$f_t^{b,thr} \leftarrow k - means(H, 3)$

$H' = \{\forall h \in H \mid \lambda_t^{m,h} \leq f_t^{b,thr}\}$

if panne de disque avec bloc b **or** $|D^b_{t_{cur}} - D^b_{t_w} \leq D^b_{thr}|$ **then**

  $t^b_{dec} \leftarrow min(t_f, t_w)$

  // paramètres primaires et secondaires : p'_b, p''_b

  $(p'_b, p''_b) \leftarrow$Parcourir l'arbre de décision P

  // groupe principal, groupe secondaire : g'_b, g''_b

  $g'_b, g''_b \leftarrow \emptyset$

  // disques de référence groupe primaire, secondaire : r'_b, r'_b

  $g'_{temp} \leftarrow \emptyset$, $g'_{temp} \leftarrow H$, $g''_{temp} \leftarrow \emptyset$, $g''_{temp} \leftarrow H$

  for $g \in g'_{temp}$ **do**

    Calculer la formule en (5.30) avec $h_i = r'_b$ et $h_j = g$

  end for

  for $g' \in g''_{temp}$ **do**

    Calculer la formule en (5.30) avec $h_i = r''_b$ et $h_j = g'$

  end for

  Trier g'_{temp}, g''_{temp}

  // dimensions max. groupes primaire et secondaire : $dim(g'_b)$, $dim(g''_b)$

  $i, j \leftarrow 0$

  while $i \leq dim(g'_{temp})$ **do**

    Prendre prochain élément k dans g'_{temp} et mettre dans g'_b

    $i \leftarrow i + 1$

  end while

  while $j \leq dim(g''_{temp})$ **do**

    Prendre prochain élément k dans g''_{temp} et mettre dans g''_b

    $j \leftarrow j + 1$

  end while

  $I \leftarrow g'_b \cup g''_b$

end if

return I

5.2.3 Description des paramètres de l'heuristique

Après avoir ainsi réduit l'espace de recherche originel, en retenant les profils de disques durs les plus susceptibles de faire partie de la solution optimale, nous laissons à un algorithme génétique le soin de faire l'exploration des solutions potentielles. C'est dans ce cadre que nous expliquons ci-bas les détails des opérateurs génétiques utilisés.

- **Sélection**

Le processus de sélection consiste à choisir parmi les individus présents dans la population initiale, ceux ayant la plus forte probabilité de contribuer à l'obtention de meilleurs solutions au fil des générations à travers le processus de croisement et de mutation.

Au début, Heron génère une population initiale composée de schémas candidats d'allocation de réplicas construits en effectuant une sélection aléatoire de disques durs au sein de l'espace de recherche réduit jusqu'à satisfaction des contraintes (5.2, 5.3, 5.6, 5.8 et 5.9) pour chaque schéma candidat d'allocation de réplicas.

Il est également impératif de tenir compte de la fluctuation de la demande des requêtes d'accès aux données et la variabilité de celles-ci. Cependant, satisfaire cette exigence en tout temps aurait requis que conformément à la variation de la demande on ajoute ou supprime des réplicas. Cela nuit bien évidemment à la stabilité globale du système.

Dès lors, au sein de Heron, nous prédisons la tendance de la demande des requêtes d'accès en utilisant la technique de prévision ARIMA. Nous nous assurons alors que l'horizon de prédiction (c.à.d. l'étendue de la prédiction en avance) est assez proche par rapport au temps présent pour conserver une précision acceptable et éviter un sur-approvisionnement prématuré de réplicas pouvant ainsi occasionner

des surcoûts.

Ainsi, soit D_t^b la demande en termes de requêtes d'accès à l'instant présent t pour un bloc de données b. Soit $\{D_{t+1}^b, D_{t+2}^b, ..., D_{t+i}^{b,h}\}$ avec i représentant l'horizon de prévision. Nous maintenons la stabilité du système face à la fluctuation de la demande en termes de requêtes d'accès aux données en prenant la demande ajustée D_{aj}^b telle que :

$$D_{aj}^b = màx(\{D_{t+1}^b, D_{t+2}^b, ..., D_{t+i}^b\}) \tag{5.31}$$

Avec la demande ajustée pour un bloc de données b, nous tenons compte des contraintes (5.8 et 5.9). Cela a, potentiellement, pour conséquence un sur-approvisionnement de réplicas temporaire sur la période de temps $\{t, t+1, ..., t+i\}$

Étant donné qu'une pré-sélection a été faite à travers le mécanisme de réduction de l'espace de recherche visant à regrouper les meilleurs individus de la population, nous considérons la sélection aléatoire comme étant raisonnable.

La complexité pour générer un schéma candidat d'allocation est $\mathcal{O}(n \cdot m)$ avec n et m dénotant respectivement la taille de l'espace de recherche réduit et le nombre de réplicas nécessaires pour satisfaire les exigences SLA.

- **Croisement**

La prochaine étape de l'exécution d'Heron consiste à effectuer des opérations de croisement de chromosomes. Le croisement est un processus de reproduction à travers lequel deux parents ayant réussi à traverser le processus de sélection génèrent un ou des enfants biologiquement mieux adaptés en combinant leurs caractéristiques. Dans le contexte d'Heron, il s'agira de combiner deux schémas d'allocations pour en construire un schéma fils d'allocation qui optimisera mieux la fonction objective. Le croisement consistera également à déterminer la manière

dont les enfants héritent des caractéristiques de leurs parents.

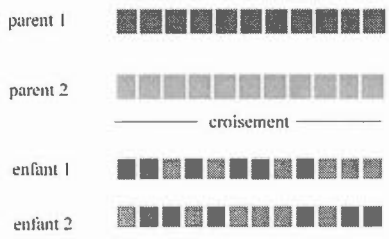

Figure 5.5 Exemple de croisement uniforme

Pour Heron, nous choisissons d'utiliser un croisement uniforme (Syswerda, 1989) tel qu'illustré à la figure 5.5 et qui consiste à déterminer de manière probabiliste si une caractéristique (plus précisément un disque dur servant d'emplacement de réplicas) d'un parent devrait être hérité par un enfant. L'avantage du croisement uniforme est qu'il nous permet de maximiser la meilleure information en provenance de l'un des deux parents.

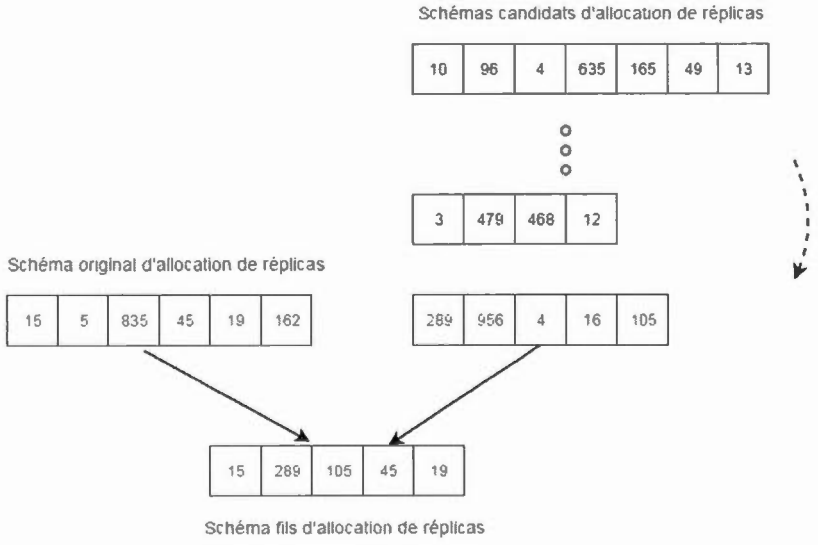

Figure 5.6 Croisement des schémas d'allocation de réplicas.

Notre stratégie de croisement consiste à toujours retenir le schéma d'allocation

précédemment appliqué pour le bloc de données — que nous appelons pour la suite schéma original d'allocation — comme étant un des parents du processus de croisement. Nous prenons alors successivement comme autre parent un schéma candidat d'allocation auparavant généré pour générer un schéma fils d'allocation comme illustré à la figure 5.6.

L'objectif global de notre stratégie de croisement est de nous assurer qu'entre le schéma original d'allocation et le schéma fils d'allocation, il existe au moins un certain niveau de similitude, et ce pour deux raisons : la première est que cela évite d'effectuer simultanément un nombre élevé d'opérations de réplication ; la seconde raison est que cela permet d'assurer la stabilité globale du système améliorant la gestion de la cohérence des réplicas.

Évidemment, les schémas d'allocation n'ont pas les mêmes tailles du fait que les disques sont hétérogènes. Ainsi, le nombre nécessaire de réplicas pour satisfaire les exigences SLA varient.

La complexité temporelle pour construire un schéma fils d'allocation est $\mathcal{O}(n)$ avec n dénotant la taille maximale entre les deux parents.

- **Mutation**

Le rôle de cet opérateur est d'introduire un aléa, permettant une exploration efficiente de nouvelles régions de l'espace de recherche avec comme objectif d'éviter une convergence prématurée ainsi que les optimums locaux (Mitchell, 1996) en s'assurant qu'il y ait un minimum de diversité génétique au sein de la population. Le principe de la mutation est de modifier aléatoirement une portion de bits ou un groupe de gènes du chromosome de sorte à générer un nouvel individu. Effectivement, un des problèmes du croisement est que le schéma fils d'allocation généré peut ne pas être un schéma valide (c.à.d violant par exemple des contraintes

de placement des réplicas). Ainsi, après le croisement, un schéma fils d'allocation non valide est susceptible de subir une mutation de certaines caractéristiques afin de le corriger en fonction de la probabilité de mutation.

On note également que la mutation ne s'applique pas forcement à tous les individus de la population. En effet, la fréquence selon laquelle les individus subissent une mutation de gènes est déterminée par la probabilité de mutation. Afin d'éviter une recherche aléatoire, la valeur de la probabilité de mutation retenue (de manière statique ou dynamique) doit rester relativement basse (typiquement moins de 2%).

Nous procédons ensuite à déterminer la qualité des schémas fils d'allocation à travers une fonction évaluation correspondant à l'équation 5.17. Dès lors, nous trions chaque schéma fils d'allocation par ordre décroissant en fonction de son score d'évaluation.

Afin d'accélérer l'exécution d'Heron et éviter les optimums locaux, nous plaçons temporairement les disques durs apparaissant régulièrement au sein des schémas fils d'allocation mal classés dans une liste noire de sorte à ce qu'ils n'apparaissent plus au sein de schémas fils d'allocations ultérieurs.

À l'issue de la phase de mutation, les schémas d'allocation infaisable (*infeasible*) sont éliminés et une portion des schémas fils d'allocation ayant les meilleurs scores d'évaluation sont portés à la prochaine itération de sélection, croisement et mutation. Ce processus — appelé élitisme — a pour objectif de s'assurer qu'à chaque nouvelle itération, les solutions potentielles s'améliorent globalement. La condition d'arrêt des itérations est définie par le temps de décision du bloc de données t^b_{dec}.

À la fin des itérations, le schéma d'allocation ayant le meilleur score d'évaluation est adopté comme solution finale pour le bloc de données concerné. Si la dimension du schéma original d'allocation est plus petite que celle du schéma adoptée d'allo-

cation, une migration de réplicas s'effectue avant la création des réplicas restants du schéma adopté. À l'inverse, si la dimension du schéma original d'allocation est plus petite que celle du schéma adopté d'allocation, une migration des réplicas s'effectue entre les deux schémas avant que les réplicas du schéma original soient supprimés. Nous résumons la stratégie décrite ci-dessus à l'algorithme 2.

Algorithme 2 Algorithme génétique

Entrées : Population réduite de disques durs $I = \{1, 2, ..., |I|\}$, Schéma d'allocation original S^b_{ori}, t^b_{dec}

Sortie : Schéma d'allocation alternatif S^b_{alt}

1: $t_{cur} \leftarrow$ temps courant

// Ensemble de schémas issus de l'élitisme : $P^b_{eli} \leftarrow \emptyset$

// Dimension max de P^b_{eli} : $|P^b_{eli}|$

2: **while** $t_{cur} \leq t^b_{dec}$ **do**

// Population initiale : $P^b \leftarrow \emptyset$

// Dimension max. de P^b : $|P^b|$

Sélection

3: $j \leftarrow 0$

4: $D^b_{aj} = max(\{D^b_{i+1}, D^b_{i+2}, ..., D^b_{i+o}\})$

// Schéma candidat d'allocation : S^b_{can}

5: $S^b_{can} \leftarrow \emptyset$

6: **while** $j \leq |P^b|$ **do**

7: **while** contraintes (5.6), (5.8) et (5.9) non respectées avec $D^b_i = D^b_{aj}$ **do**

8: $disqueTemp \leftarrow rand(i \mid i \in I)$

9: **if** contraintes (5.2), (5.3) respectées **then**

10: $S^b_{can} \leftarrow S^b_{can} \cup disqueTemp$

11: **end if**

12: **end while**

13: $P^b \leftarrow P^b \cup S^b_{can}$

14: $S^b_{can} \leftarrow \emptyset$

15: $j \leftarrow j + 1$

16: **end while**

Croisement

// Ensemble de schémas fils : $P^b_{fils} \leftarrow \emptyset$

17: **for** $p^b \in P^b$ **do**

// Schémas fils d'allocation : S^b_{enf}, S^b_{enf1}

18: $(S^b_{enf}, S^b_{enf1}) \leftarrow$ croisementUniforme(S^b_{ori}, p^b) - (Syswerda, 1989)

19: $P^b_{fils} \leftarrow P^b_{fils} \cup (S^b_{enf}, S^b_{enf1})$

20: **end for**

21: $P^b_{fils} \leftarrow P^b_{fils} \cup P^b_{eli}$

Mutation

// *Ensemble des schémas fils invalides :* P^b_{inv}

22: $P^b_{inv} \leftarrow P^b_{inv} \cup \{p^b \in P^b_{fils} \mid p^b$ ne respecte pas contrainte (5.2) $\}$

// *probabilité de mutation :* m

23: Selon m, modifier $p^{'} \in P^b_{inv}$

24: Calculer la fonction d'évaluation, eval en 5.17 pour $\{p^b \in P^b_{fils}\}$ et trier P^b_{fils}

25: **for** $k = 0; k < |P^b_{eli}|; k++$ **do**

26: Prendre prochain élément dans P^b_{fils} trié

27: **end for**

28: $t_{cur} \leftarrow$ temps courant

29: **end while**

30: $S^b \leftarrow min[\ eval(p^b)\ \forall p^b \in P^b_{fils}]$

31: **return** S^b

CHAPITRE VI

ÉVALUATION DES PERFORMANCES DE HERON

Dans cette section, nous comparons les performances de notre solution, Heron, décrite auparavant, avec deux approches de gestion de réplicas largement utilisées dans la littérature et qui ne tiennent pas compte de l'hétérogénéité des taux de pannes pour les disques durs au sein de l'infrastructure de stockage infonuagique.

6.1 Environnement de simulation

Pour les besoins de nos simulations, nous implémentons Heron à travers un outil de simulation d'infrastructure infonuagique dénommé CloudSim (Calheiros *et al.*, 2011).

Ainsi, nous créons trois fichiers traces dans lesquels nous renseignons les paramètres de notre environnement de simulation, à savoir, les requêtes d'accès aux données initiales, les spécifications des disques durs composant notre infrastructure de stockage infonuagique ainsi que les paramètres relatifs à l'exécution de l'algorithme génétique.

Également, nous créons une nouvelle classe *HardDrive* en étendant la classe CloudSim *Storage.java*. Pour prendre en compte l'hétérogénéité, nous spécifions dans notre classe le modèle du disque à la création d'une nouvelle instance.

Nous effectuons également les extensions des classes CloudSim suivantes pour programmer les modules de notre plateforme de gestion :

- *Cloudlet.java* pour l'envoi des requêtes d'accès en provenance des clients ;
- *PowerDatacenterBroker.java* pour jouer le rôle du module proxy ;
- *Datacenter.java* pour jouer à la fois le rôle du module SAP et de gestion ;
- *SimEntity.java* pour créer une nouvelle entité capable d'effectuer le rôle du module de mappage entre blocs L-à-P.

Nous modélisons la demande concernant les requêtes d'accès aux fichiers ainsi que leurs tailles en nous basant sur les fichiers de trace fournis par Yahoo ! (Yahoo, 2015). Nous l'adaptons à notre infrastructure de stockage infonuagique en utilisant la méthode d'échantillonage stratifié (*stratified sampling*) (Thompson, 2012). Nous nous assurons que la variabilité de la demande d'accès aux données au fil du temps en distribuant le temps d'arrivées entre requêtes d'accès selon un mélange de distributions exponentiel, pareto et weibull. Nous fixons l'horizon de prédiction à deux périodes de temps correspondant à environ 6 jours.

Nous considérons qu'au sein de notre infrastructure de stockage infonuagique, tous les fichiers stockés sont mappés à un ensemble de blocs de données de taille fixe auxquels sont assignés des exigences SLA. Par souci de simplicité, lorsque le client du stockage infonuagique assigne des exigences SLA à un fichier, nous considérons que tous les blocs de données constituant ce fichier requièrent également les mêmes exigences SLA.

Soit alors deux type d'exigences : la disponibilité minimale et le temps de réponse maximum suite à une requête d'accès. Nous assignons la disponibilité et le temps de réponse requis en choisissant aléatoirement une valeur respectivement dans les ensembles {99.0, 99.9, 99.99, 99.999, 99.9999} et {0.1,0.2,0.5,1} (s). Initialement

dans l'infrastructure de stockage, nous disposons de 100000 blocs de données dont nous voulons garantir la satisfaction des exigences SLA tout en minimisant les coûts liés à la gestion des réplicas.

Modèle	Cap. (TB)	Puiss. Act. (W)	Taux de pannes (%)		
			Année 1	Année 2	Année 3
WD10EADS	1.0	5.4	4.29	3.90	9.91
WD10EACS	1.0	7.5	0.01	5.21	0.01
ST31500541AS	1.5	5.8	10.52	9.52	12.07
ST31500341AS	1.5	11.6	23.29	23.53	26.29
HDS722020ALA330	2.0	9.4	1.03	1.07	2.81
DT01ACA300	3.0	6.4	6.93	3.68	2.80
WD30EFRX	3.0	4.4	3.79	6.94	8.79
ST33000651AS	3.0	9.3	6.91	4.80	3.55
STM3000DM001	3.0	8.0	10.35	43.08	30.94
HDS723030ALA640	3.0	8.6	1.01	2.27	2.12
HDS5C3030ALA630	3.0	6.4	0.99	0.59	1.31
HMS5C4040ALE640	4.0	6.2	3.85	1.41	0.70
HDS5C4040ALE630	4.0	6.0	1.65	0.91	0.86
ST4000DX000	4.0	7.5	1.12	1.12	3.73
ST4000DM000	4.0	7.5	4.17	2.58	3.31

Tableau 6.1 Modèles de disques dans l'infrastructure de stockage infonuagique.

Afin de modéliser une infrastructure de stockage hétérogène, nous simulons 5000 disques durs de modèles et fournisseurs différents. Nous rapportons, au tableau 6.1, les caractéristiques pertinentes pour chaque modèle de disque dur. Nous reprenons ici les données fournies par (Backblaze, 2015) et provenant de l'évaluation de plus de 50000 disques durs.

Le taux de pannes seuil utilisé dans le mécanisme de réduction de l'espace de recherche est fixé à 2.37% correspondant au centroïde de plus petite valeur du k-moyennes (*k-means clustering*) avec k = 3 appliqué sur les donnees du Tableau 6.1.

À des fins réalistes, nous configurons le taux de pannes annuel de chaque disque dur dans un intervalle de ±20% de celui estimé du modèle de disque. Ainsi, en fonction de la valeur du taux de pannes annuel, chaque disque dur est susceptible de tomber en panne tout au long de l'année. Avant qu'une panne de disque se produise, elle est détectée, confirmée, et une notification est envoyée trois (3) jours à l'avance au module de supervision, d'analyse et de prédiction. Cela est une hypothèse raisonnable dans la mesure où, selon (Li *et al.*, 2014), il est possible de maintenir une prédiction à un niveau de précision de 96% et une notification en avance de 352 heures. Le module de gestion est alors chargé de trouver les schémas d'allocation de réplicas alternatives pour tous les blocs de données affectés par la panne du disque dur dans l'optique de continuer à garantir les exigences SLA.

Afin d'étudier l'impact de l'hétérogénéité de l'infrastructure de stockage, nous considérons, dans nos simulations, trois scénarios différents illustrés dans le tableau 6.2. De scénario en scénario, nous réduisons la proportion de disques durs premium dans l'infrastructure de stockage infonuagique tout en augmentant le nombre de disques durs ayant des taux de pannes élevés.

À des fins de comparaisons, nous évaluons l'efficacité d'Heron avec deux approches — statique et glouton — que nous résumons à travers les algorithmes 3 et 4. Notre objectif ici, vu qu'il existe peu d'études sur la disponibilité dans une infrastructure de stockage infonuagique, est de comprendre l'influence de l'hétérogénéité lorsqu'on se focalise sur le nombre des réplicas (dans le cas de l'approche statique) ou l'emplacement des réplicas (dans le cas de l'approche gloutonne).

Tableau 6.2 Scénarios de simulation

Scénario 1

Modèle	# de disques
WD10EADS	125
ST31500341AS	130
ST31500541AS	370
HDS722020ALA330	1500
ST33000651AS	375
HDS5C3030ALA630	2500

Scénario 2

Modèle	# de disques
ST31500541AS	380
ST3000DM001	120
WD30EFRX	1500
HMS5C4040ALE640	2100
ST4000DM000	580
ST4000DX000	1400

Scénario 3

Modèle	# de disques
ST31500341AS	630
HDS723030ALA640	1220
ST3000DM001	370
DT01ACA300	150
WD30EFRX	850
HMS5C4040ALE640	1780

Algorithme 3 Stratégie statique

Input : blocs de données $B = \{1, 2, ..., |B|\}$

Output : Schémas d'allocations statiques

1: **for** $b \in B$ **do**
2: Choisir aléatoirement 3 disques durs distincts.
3: Placer réplica de b.
4: **end for**

L'approche statique consiste sommairement à créer, pour chaque bloc de données, trois réplicas que l'on place sur trois disques durs sélectionnés aléatoirement. Par souci d'équité, nous nous assurons que les réplicas sont placés dans des disques

Algorithme 4 Stratégie gloutonne

Input : bloc de données $b \mid b \in B = \{1, 2, ..., |B|\}$, m

Output : Schémas d'allocation

1: Choisir initialement un groupe de disque G jusqu'à satisfaction des exigences SLA
2: Récupérer pour chaque disque dur h, le taux de pannes $\bar{\lambda}^h$ à l'instant t actuel
3: Récupérer pour chaque disque dur h, le temps de service $\bar{s}^h$
// Arrêt du boucle : m
4: **for** $i = 0; i < m; i++$ **do**
5: **for** $g \in G$ **do**
6: Trier selon $\bar{\lambda}^g$
7: Choisir aléatoirement un disque dur f
// Fonction utilité : $u()$
8: Calculer $u(\bar{\lambda}^g,\ \bar{s}^g)$ et $(\bar{\lambda}^f,\ \bar{s}^f)$
9: **if** $u(\bar{\lambda}^g,\ \bar{s}^g) \geq (\bar{\lambda}^f,\ \bar{s}^f)$ **then**
10: Remplacer g par f
11: **end if**
12: **end for**
13: **end for**

durs distincts.

L'approche gloutonne, quant à elle, lorsqu'une violation des exigences SLA est détectée, s'applique à d'abord choisir aléatoirement un groupe de disques durs satisfaisant les exigences SLA. Ensuite, au fil des itérations, les membres du schéma d'allocation du bloc de données sont progressivement remplacés par des disques durs ayant une meilleure utilité. La fonction utilité calcule la contribution apportée par l'ajout d'un disque selon deux paramètres : le taux de pannes actuels et le temps de service. On peut aisément voir que cette fonction cherche typiquement, les disques durs ayant un taux de pannes et un temps de service bas. Nous arrêtons les itérations de l'approche gloutonne à 50.

6.2 Résultats

Afin d'évaluer la pertinence de notre solution, nous considérons six (6) métriques : le taux de violation en termes de disponibilité, le taux de violation en termes de temps de réponse suite à une requête d'accès aux données, le nombre de réplicas créés, le nombre de migration effectuées, la consommation énergétique totale et le coût à la suite de violations en termes de disponibilité.

Une violation de disponibilité se produit seulement lorsque tous les réplicas sont indisponibles lors d'une requête d'accès aux données concernées à cause notamment de pannes de disques durs. Une violation de temps de réponse intervient lorsque, suite à l'émission d'une requête d'accès à un bloc, le système de stockage infonuagique retourne les données au-delà du temps indiqué lors de la spécification des exigences SLA.

Figure 6.1 Violation en termes de disponibilité.

Globalement, on observe qu'à travers nos scénarios de simulation, Heron induit moins de violations en termes de disponibilité que l'approche statique et celle

gloutonne. Nous illustrons cela à la figure 6.1. Heron tire profit de sa capacité à prédire les pannes de disques durs pour éviter une indisponibilité conjointe des réplicas. De plus, Heron, à cause du sur-approvisionnement de réplicas pour lisser la variation de la demande en termes de requêtes d'accès, améliore la disponibilité.

Il est intéressant de noter que dans les scénarios où il y a une majorité de disques durs premium (scénarios 1 et 2), la méthode statique fournit des résultats concurrençant ceux de Heron et de la méthode gloutonne laissant supposer qu'une stratégie consistant à simplement ajouter des réplicas sans se soucier de leur emplacements pourrait etre convenable dans un contexte d'hétérogénéité contrôlée (à savoir que la plupart des disques durs ont de très faibles taux de pannes). Toutefois, on constate également que cette stratégie atteint ses limites dans un scénario (scénario 3) où il y a une plus basse majorité de disques durs premiums. En effet, dans ce scénario, il existe un plus forte probabilité que les disques durs contenant les réplicas tombent en panne dans un intervalle très proche.

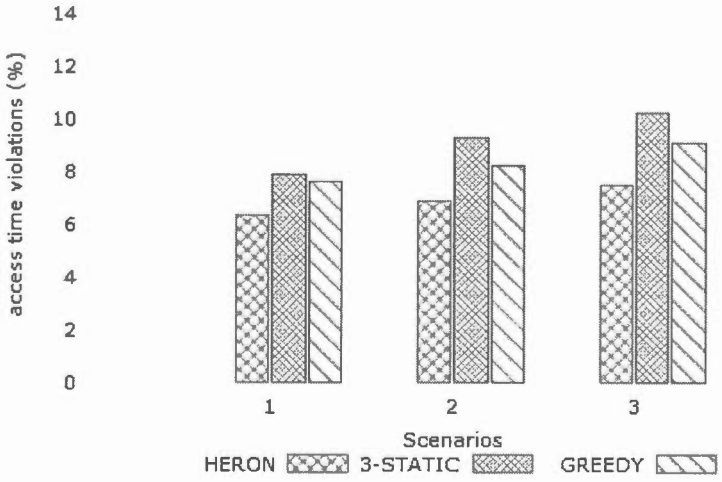

Figure 6.2 Violation en termes de temps de réponse suite à une requête d'accès.

Cela explique également les résultats présentés à la figure 6.2. On notera aussi

que le taux de violations en termes de temps d'accès est plus élevé que le taux de violations en termes de disponibilité. On explique cela par le fait qu'une panne de disque dur n'engendre pas forcement une violation des exigences en termes de disponibilité, tant qu'il demeure d'autres réplicas. Cependant, relativement au temps de réponse, les réplicas jouent également un rôle d'équilibrage de la charge. Ainsi, une panne peut entraîner une violation de temps de réponse pour certains blocs de données subissant une demande et une variabilité forte. Bien que Heron permette de lisser la fluctuation de la demande, elle implique également une création de réplicas plus élevée par rapport à l'approche gloutonne telle qu'illustré à la figure 6.3. Par conséquent, la consommation énergétique est également plus forte pour Heron par rapport à la méthode gloutonne. Évidemment, cela est un compromis nécessaire pour tenir en compte de la variabilité de la demande de requêtes d'accès.

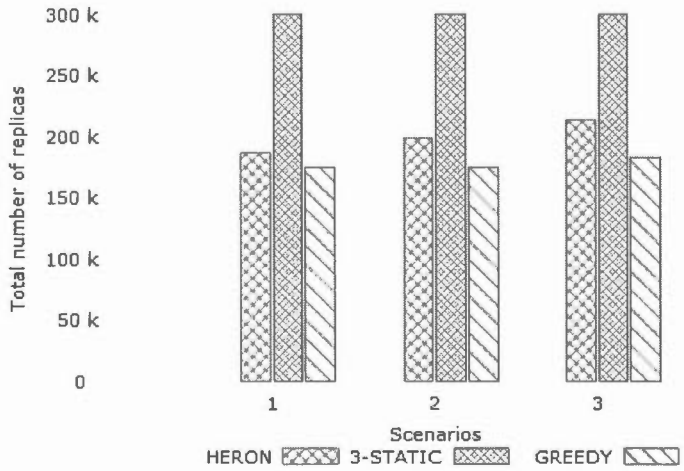

Figure 6.3 Nombre de réplicas créés.

Par ailleurs, nous observons à la figure 6.5 que Heron réussit à réduire de façon substantielle le nombre de migrations effectuées lors de l'instauration des schémas d'allocation de réplicas en comparaison avec l'approche gloutonne. La différence se

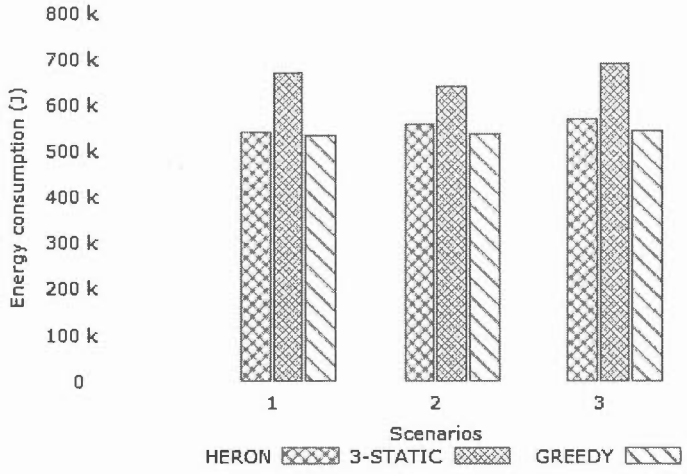

Figure 6.4 Énergie totale consommée.

situant dans le fait qu'Heron, à travers son opérateur de croisement, s'évertue à ce que d'un schéma d'allocation à un autre, on retient des emplacements de réplicas du schéma original d'allocation si possible, de sorte à minimiser le nombre de migrations.

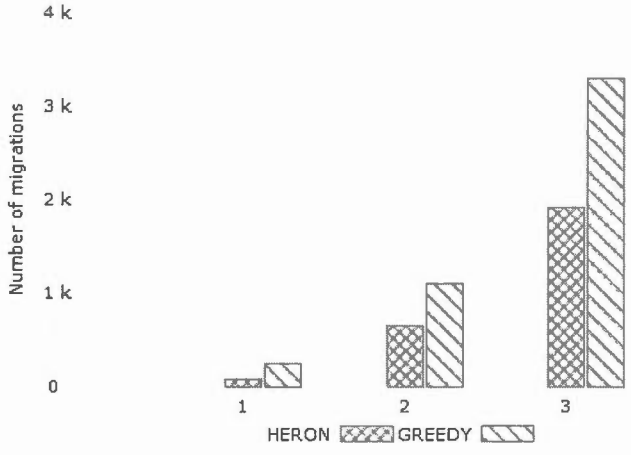

Figure 6.5 Nombre total de migrations effectuées.

A l'inverse, l'approche gloutonne peut engendrer d'un schéma d'allocation à un

autre, un changement complet d'emplacements de réplicas résultant en un nombre significatif de migrations de réplicas.

Enfin, un autre aspect que nous considérons, pour mieux distinguer les avantages d'Heron par rapport à l'approche gloutonne, consiste à appliquer un mécanisme de pénalisation monétaire dans lequel nous appliquons un coût de plus en plus croissant en accord avec la sévérité des exigences SLA en termes de disponibilité.

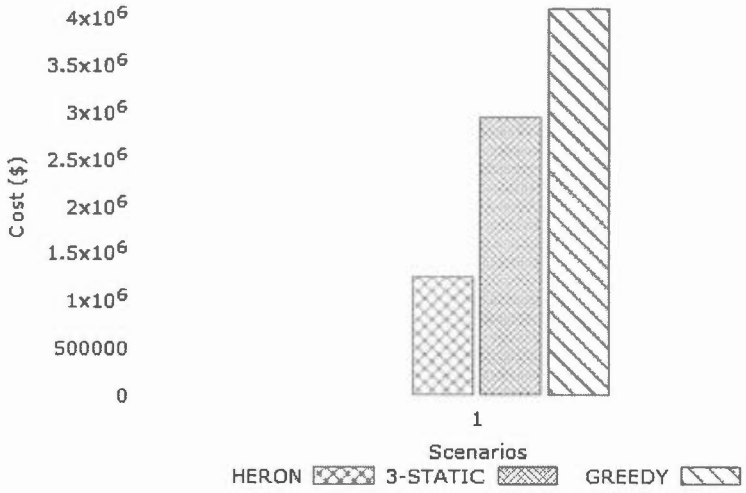

Figure 6.6 Coût suite aux violations en termes de disponibilité.

À la figure 6.6, nous montrons les coûts pour le pire scénario de simulation (scénario 3). Nous remarquons qu'en comparaison à d'autres approches, Heron réussit maintenir un bas coût à la suite de violations en termes de disponibilité. Cela est une conséquence du mécanisme de réduction de l'espace de recherche dans lequel les blocs de données ayant des exigences SLA strictes obtiennent un espace de recherche réduit composé principalement de disques durs premium et stables.

CHAPITRE VII

CONCLUSION

L'hétérogénéité de l'infrastructure de stockage infonuagique, plus particulièrement, son impact par rapport à la disponibilité des données, a été largement ignorée par les études précédentes. Pourtant, de nombreuses études récentes ont indiqué que d'un modèle de disque à un autre, les fréquences de pannes mesurées par le taux de pannes variaient significativement, mais qu'en plus ce taux était variable au fil du temps. On se pose donc la question à savoir quel est l'impact de ces phénomènes dans un modèle de gestion des données faisant usage de la réplication pour garantir la disponibilité des données ?

Dans ce cadre, nous avons étudié la problématique de la gestion des réplicas dans le cadre d'une infrastructure de stockage infonuagique hétérogène. Notre objectif consistait à pouvoir garantir la satisfaction des exigences SLA tout en minimisant les coûts de gestion induits, à la fois par la variabilité de la demande en termes de requêtes d'accès aux données des utilisateurs mais également la variation des taux de pannes des disques durs au fil du temps. Ainsi, nous avons formulé la problématique sous forme d'un problème d'optimisation et nous avons montré qu'il faisait partie des problèmes NP-difficile, de par le fait qu'il s'agissait d'une généralisation du «bin packing problem».

À partir de là, nous avons proposé une plateforme de gestion de réplicas dans

laquelle figure une heuristique dénommée Heron et qui se base sur un algorithme génétique pour chercher une solution stable et proche de l'optimal pour chaque bloc de données dans le système de stockage.

La performance de Heron a été évaluée et comparée à deux modèles de gestion de réplicas fréquemment utilisés dans la littérature : une approche statique et une approche gloutonne, et ce pour nous permettre de situer l'impact de l'hétérogénéité sur les objectifs SLA. Les résultats montrent que l'approche statique était capable de concurrencer l'approche gloutonne et Heron lorsque l'hétérogénéité en termes de taux de pannes demeure relativement faible. Cependant, sa consommation énergétique ainsi que l'espace de stockage consommée reste un point faible. Également, dans nos simulations, l'approche gloutonne parvient à obtenir de meilleurs résultats que Heron en matière de consommation énergétique, du fait que Heron effectue du sur-approvisionnement de réplicas pour lisser les demandes d'accès aux données futures. Toutefois, globalement, Heron obtient de meilleurs résultats en garantissant un taux moindre de violations en termes de disponibilité, de migrations effectuées.

Nous envisageons, dans de futurs travaux, d'adapter à notre modèle plusieurs contributions préalablement existantes dans la littérature tout en gardant l'équité nécessaire en termes de comparaison. Nous tirerons également avantage de plus de données, (complètes et récentes) concernant les pannes de disques durs, en provenance de Backblaze.

Nous comptons utiliser ces travaux pour améliorer notre plateforme et ainsi avoir un regard plus approfondi sur l'impact de l'hétérogénéité des disques durs dans la satisfaction des SLA.

APPENDICE A

MÉCANISME DE PRÉVISION DE LA DEMANDE : ARIMA

La prévision de la demande s'effectue à travers une série temporelle consistant à analyser mathématiquement le comportement passé d'un phénomène afin d'en déduire avec la meilleure précision possible le comportement futur (Zhani, 2011).

Dans le cadre de notre travail, nous nous intéressons à prédire la demande liée aux requêtes d'accès aux données en provenance de clients du système de stockage infonuagique. Il est important de noter que dans le cadre de notre étude, nous focalisons surtout notre attention à détecter une tendance claire concernant la demande, plus particulièrement croissante, de facon à éviter une violation par rapport au temps de réponse exigé pour les données.

Ainsi, on considère un modèle de prévision représenté par une fonction f et dont les anciennes observations de la demande concernant une donnée sont définies à travers une série temporelle $y(t),\ y(t-1),\ y(t-2), ...,\ y(t-n)$ avec n, le nombre d'observations antérieures. La valeur prédite $\hat{y}(t+h)$ à un horizon prévisionnel h est définie telle que :

$$\hat{y}(t+h) = f(\ y(t),\ y(t-1),\ y(t-2), ...,\ y(t-n)\ ) \tag{A.1}$$

Pour la suite de notre analyse, considérons un modèle de prévision particulier dénommé ARIMA (*AutoRegressive Integrated Moving Average*) et que nous uti-

lisons dans le cadre de notre travail. La raison est que dans un environnement infonuagique, la méthode ARIMA a été démontré comme étant celle fournissant les meilleurs résultats en termes de précision dans le cadre de la prédiction des requêtes de ressources (Calheiros *et al.*, 2015; Zhao *et al.*, 2012; Zhang *et al.*, 2012; Zhang *et al.*, 2013; Vakilinia *et al.*, 2016). À cet égard, (Calheiros *et al.*, 2015) estiment la précision de prédiction d'ARIMA jusqu'à 91% dans son analyse des traces réels de requêtes en provenance de serveurs web. Dans la même veine, (Zhang *et al.*, 2013) ont analysé, à travers ARIMA, des traces réelles en provenance des clusters de traitement Google (*Google compute clusters*) pour prédire les taux d'arrivée ainsi que le nombre de tâches. Ils indiquent que l'erreur quadratique moyenne des prédictions (*root square error*) était inférieure à 1%.

Supposons, donc, le modèle ARIMA(p,d,q) pour lequel les variables p, d et q désignent respectivement le nombre de termes auto-régressifs, le nombre de différenciations et le nombre de moyennes mobiles. Ainsi, le modèle ARIMA combine, dès lors, trois modèles : un modèle d'auto-régression, un modèle de différenciation ou d'intégration, et un modèle à moyenne mobile.

L'auto-régression (*AutoRegressive - AR*) peut être défini comme étant un modèle dans lequel la valeur du résultat dépend linéairement d'un ensemble de valeurs précédentes et d'une expression d'erreur stochastique. Un processus à moyenne mobile (*Moving Average - MA*), quant à lui, estime que la valeur du résultat est linéairement dépendante de l'expression d'erreur stochastique courante ainsi que des expressions d'erreurs précédentes.

La combinaison des modèles AR et MA constitue le modèle auto-régressif à moyenne mobile (*AutoRegressive Moving Average - ARMA*). Formellement, une série temporelle $y(t)$ suit un processus $ARMA(p, q)$ si elle est stationnaire et si

pour chaque instant t :

$$y(t) = \phi_1 \cdot y(t-1) + ... + \phi_p \cdot y(t-p) + \epsilon(t) + \theta_1 \cdot \epsilon(t-1) + ... + \theta_q \cdot \epsilon(t-q) \quad \text{(A.2)}$$

avec ϕ_i et θ_j désignent des valeurs réels et $\epsilon(t)$, dénotant un bruit blanc de variance σ^2.

Nous reformulons l'équation A.2 de manière plus concise :

$$y(t) = \sum_{i=1}^{p} \phi_i \cdot L^i \cdot y(t) + (1 + \sum_{i=1}^{q} \theta_i \cdot L^i) \cdot \epsilon(t) \quad \text{(A.3)}$$

où L dénote l'opérateur "retard" (*lag operator*). Le modèle ARMA emploie p+q+1 paramètres inconnus déterminés à travers les fonctions d'auto-corrélation (ACF) et d'auto-corrélation partielle (PACF). Les paramètres ϕ_i et θ_i peuvent être estimés grâce à la méthode *d'estimation de la plausibilité maximale* (*Maximum Likelihood - MLE*) (Aldrich *et al.*, 1997).

On notera qu'il est possible qu'une série temporelle viole l'hypothèse de stationnarité à cause des variations. Or, la stationnarité est un paramètre essentiel dans la détermination des modèles ARIMA. Autrement dit, la variance ainsi que la valeur moyenne de la série temporelle sont constantes au fil du temps. Ainsi, on procède à la stationnarisation de la série temporelle à travers une différenciation d'ordre d, obtenant ainsi une version *intégrée (Integrated - I)* de la série.

Le modèle ARIMA constitue donc une version intégrée du modèle ARMA. Ainsi, une série temporelle $y(t)$ suit un processus ARIMA lorsque l'expression $(1 - L)^d \cdot y(t)$ suit un processus $ARMA(p, q)$ avec d, une valeur entière. Nous avons donc le modèle $ARIMA(p, d, q)$ tel que :

$$(1 - \sum_{i=1}^{p} \phi_i \cdot L^i) \cdot (1 - L)^d \cdot y(t) = (1 + \sum_{i=1}^{q} \theta_i \cdot L^i) \cdot \epsilon(t) \quad \text{(A.4)}$$

APPENDICE B

MÉCANISME DE PRÉVISION DES PANNES DE DISQUES DURS

La prédiction des pannes au niveau des disques durs a suscité, ces dernières années, de nombreuses études (Murray *et al.*, 2005; Hamerly et Elkan, 2001; Hughes *et al.*, 2002; Murray *et al.*, 2003; Zhao *et al.*, 2010; Xu *et al.*, 2016; Wang *et al.*, 2011), s'évertuant principalement à exploiter les données S.M.A.R.T (*Self-Monitoring, Analysis and Reporting Technology*) présentes sur la plupart des disques durs modernes afin de bâtir des modèles de prédictions de pannes. Essentiellement, les données S.M.A.R.T permettent de collecter en permanence diverses informations ayant trait à la santé du disque dur telles que la température, le nombre de secteurs défectueux, le taux d'erreur lors de la lecture d'un données, etc.

Cependant, les résultats obtenus par les études auparavant citées étaient généralement modestes (moins de 60% en taux de prédiction correctes pour un taux de fausses alertes basses) et présentaient diverses difficultés dans la mesure où les méthodes utilisées ne permettaient pas d'avoir des performances stables en termes de prédiction (Li *et al.*, 2014).

C'est dans ce cadre que nous basons notre étude sur la base du modèle de prédiction de pannes de disques durs fourni par (Li *et al.*, 2014) qui se base principalement sur les arbres de classification et de régression pour obtenir un taux de prédiction correcte de pannes autour de 95% pour un taux de fausses alertes de

0.1% environ.

L'avantage par rapport aux autres techniques est que ces arbres de décisions constituent des modèles ayant une performance et précision hautement fiables (Li *et al.*, 2014). Ainsi, ces arbres de décisions permettent non seulement de prédire adéquatement les panne de disques durs mais également d'évaluer numériquement la santé de ceux-ci.

Nous l'utilisons donc dans notre travail afin de pouvoir prédire les pannes de disques durs et prendre ainsi des mesures correctives permettant de garantir que les données stockées sur les disques concernés puissent demeurer disponibles.

Nous présentons, ici, le fonctionnement de ce modèle de prédiction qui utilise en réalité deux types d'arbres de décision : un arbre de classification servant à déterminer si le disque dur tombera en panne et un arbre de régression dont l'objectif est de quantifier l'état de santé des disques durs.

— **Le modèle d'arbre de classification**

La classification du disque dur se fait à travers un arbre de décision dans lequel chaque noeud dénote la probabilité qu'un disque dur tombe en panne selon la valeur S.M.A.R.T observée au niveau du disque dur tel qu'illustré à la figure B.1.

L'arbre de classification utilise le gain d'information comme critère de scission à chaque noeud, consistant à fouiller à travers l'ensemble des attributs S.M.A.R.T pour trouver celui qui permet de maximiser le gain d'informations.

Ainsi, soit un noeud D scindé en deux noeuds enfants D_1 et D_2 selon un attribut S.M.A.R.T, v_i. Le gain d'information à travers cette scission peut être déterminé à partir de la formule suivante :

$$gain(D, v_i) = info(D) - info(D, v_i) \tag{B.1}$$

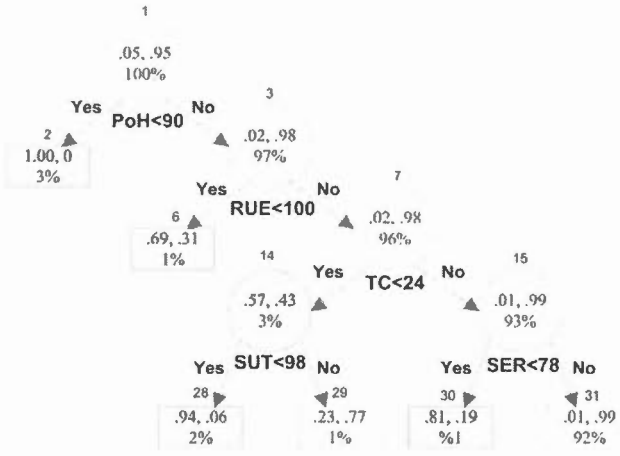

Figure B.1 Exemple d'arbre de classification pour la prédiction des pannes de disques durs (Li *et al.*, 2014).

où $info(D)$ représente l'entropie d'information au niveau du noeud D et $info(D, v_i)$, la somme des entropies d'informations des noeuds enfants après scission selon l'attribut S.M.A.R.T v_i.

Le concept d'entropie d'information sert à dénoter le biais du jeu de données par rapport à une valeur objectif et se définit par la relation suivante :

$$info(D) = -plog_2(p) - qlog_2(q) \tag{B.2}$$

avec $p, q \mid p+q = 1$ dénotant les probabilités de classification des échantillons considérés pour le noeud D. On définit la somme des entropies d'informations des noeuds enfants de la sorte :

$$info(D, v_i) = \frac{|D_1|}{|D|} info(D_1) + \frac{|D_2|}{|D|} info(D_2) \tag{B.3}$$

où $|D|$ dénote le quantité totale d'échantillons pour le noeud D. À chaque étape de la construction de l'arbre de classification, l'algorithme de classification calcule essentiellement le gain d'information pour l'ensemble des scissions possibles avant de choisir celle fournissant le gain d'information maximal.

Afin d'améliorer la classification, (Li *et al.*, 2014) utilisent une stratégie d'apprentissage consistant à ajuster de manière itérative les probabilités de classification. De plus, afin de réduire les fausses alarmes, deux types d'erreurs, à travers un jeu de poids, sont distingués : les fausses positives et les fausses négatives, affectant notamment le choix du paramètre S.M.A.R.T choisi pour un noeud donné.

— **Le modèle d'arbre de régression**

Au-delà de l'arbre de classification qui permet uniquement de déterminer si le disque dur est prévu de tomber en panne, le modèle fourni par (Li *et al.*, 2014) permet également de quantifier l'état de santé du disque dur. Cela en partant de l'idée que la détérioration de l'état du disque dur se fait plutôt graduellement et, qu'en conséquence, un mécanisme d'évaluation de la santé du disque est essentiel.

L'arbre de régression utilise la méthode des moindres carrés comme critère de scission. Ainsi, pour chaque noeud, la scission retenue est celle ayant une somme des carrés minimale. Celle-ci se calcule à travers la formule qui suit :

$$sq = \sum_{i=1}^{n} (y_i - \bar{Y})^2 \tag{B.4}$$

avec n dénotant la quantité d'échantillons pour le noeud considéré, y_i la valeur de l'attribut S.M.A.R.T pour la i_{eme} échantillon et $\bar{Y}$ la moyenne des échantillons pour la valeur de l'attribut S.M.A.R.T considérée.

Pour chaque disque dont il est prédit qu'il tombe en panne, on quantifie la santé du disque en considérant une fenêtre de détérioration par la formule suivante :

$$h_d(i) = -1 + \frac{i}{w_d} \tag{B.5}$$

avec w_d représentant la taille de la fenêtre de prédiction et i l'échantillon considéré.

BIBLIOGRAPHIE

(2014). 2014 service availability benchmark survey. Récupéré de http://www.continuitysoftware.com/wp-content/uploads/2014/05/2014-SA-Survey-Report.pdf

(2016). How your business survival depends on disaster recovery. Récupéré de https://www.itgct.com/wp-content/uploads/2016/02/How-Your-Business-Survival-Depends-On-Disaster-Recovery.pdf

Abad, C. L., Lu, Y. et Campbell, R. H. (2011). Dare : Adaptive data replication for efficient cluster scheduling. Dans *Proceedings of the 2011 IEEE International Conference on Cluster Computing*, CLUSTER '11, 159–168., Washington, DC, USA. IEEE Computer Society.

Abadi, D. J. (2009). Data management in the cloud : Limitations and opportunities. *IEEE Data Eng. Bull.*, *32*(1), 3–12. Récupéré de http://sites.computer.org/debull/A09mar/abadi.pdf

Adams, A. et Mishra, N. (2010). User survey analysis : Key trends shaping the future of data center infrastructure through 2011. *Gartner Market Analysis and Statistics (Oct. 2010)*.

Aldrich, J. *et al.* (1997). R.A. fisher and the making of maximum likelihood 1912-1922. *Statistical Science*, *12*(3), 162–176.

Antonisse, J. (1989). A new interpretation of schema notation that overturns the binary encoding constraint. Dans *Proceedings of the Third International Conference on Genetic Algorithms*, 86–91., San Francisco, CA, USA.

Arnold, A. (2010). Assessing the financial impact of downtime - luminet. Récupéré de http://luminet.co.uk/wp-content/uploads/2014/10/assessing-the-financial-impact-of-downtime-uk.pdf

Backblaze (2015). Hard drive data sets. Accessed : 2016-03-24. Récupéré de https://www.backblaze.com/blog/hard-drive-reliability-q3-2015/

Bauer, E. et Adams, R. (2012). *Reliability and availability of cloud computing*. John Wiley & Sons.

Beath, C., Becerra-Fernandez, I., Ross, J. et Short, J. (2012). Finding value in the information explosion. *MIT Sloan Management Review*, *53*(4), 18–20. Récupéré de `http://search.proquest.com/docview/1023761998?accountid=14719`

Beecher, V., Schupmann, V. et Stern, J. (2013). Oracle database high availability overview. Récupéré de `https://docs.oracle.com/cd/e11882\_01/server.112/e17157.pdf`

Bonvin, N., Papaioannou, T. G. et Aberer, K. (2010). A self-organized, fault-tolerant and scalable replication scheme for cloud storage. Dans *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, 205–216., New York, NY, USA. ACM.

Boutaba, R., Cheng, L. et Zhang, Q. (2011). On cloud computational models and the heterogeneity challenge. *Journal of Internet Services and Applications*, *3*(1), 77–86.

Calheiros, R. N., Masoumi, E., Ranjan, R. et Buyya, R. (2015). Workload prediction using arima model and its impact on cloud applications. *IEEE Transactions on Cloud Computing*, *3*(4), 449–458.

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A. F. et Buyya, R. (2011). Cloudsim : A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw. Pract. Exper.*, *41*(1), 23–50.

Cerf, R. (1994). *Une théorie asymptotique des algorithmes génétiques.* (Thèse de doctorat). Thèse de doctorat dirigée par Berlinet, Alain Mathématiques Montpellier 2 1994. Récupéré de `http://www.theses.fr/1994MON20056`

Cidon, A., Rumble, S., Stutsman, R., Katti, S., Ousterhout, J. et Rosenblum, M. (2013). Copysets : Reducing the frequency of data loss in cloud storage. Dans *Presented as part of the 2013 USENIX Annual Technical Conference (USENIX ATC 13)*, 37–48., San Jose, CA.

Coffman, Jr., E. G., Garey, M. R. et Johnson, D. S. (1997). : chapitre Approximation Algorithms for Bin Packing : A Survey, 46–93. Boston, MA, USA : PWS Publishing Co.

Cormier, G. *Chapitre 1 : L'algorithme génétique.* Récupéré de `http://www8.umoncton.ca/umcm-cormier\_gabriel/SystemesIntelligents/AG.pdf`

Delimitrou, C. et Kozyrakis, C. (2013). Paragon : QoS-aware scheduling for heterogeneous datacenters. *SIGPLAN Not.*, *48*(4), 77–88.

Delimitrou, C., Sankar, S., Vaid, K. et Kozyrakis, C. (2011). Accurate modeling and generation of storage I/O for datacenter workloads. Dans *Proceedings of the 2nd Workshop on Exascale Evaluation and Research Techniques (EXERT)*.

Dósa, G. (2007). *The Tight Bound of First Fit Decreasing Bin-Packing Algorithm Is FFD(I) 11/9OPT(I) + 6/9*, 1–11. Springer Berlin Heidelberg : Berlin, Heidelberg

Doughty, K. (2000). *Business continuity planning : protecting your organization's life*. CRC Press.

Elerath, J. (2000). Specifying reliability in the disk drive industry : No more mtbf's. Dans *Reliability and Maintainability Symposium, 2000. Proceedings. Annual*, 194–199.

Elerath, J. G. et Shah, S. (2004). Server class disk drives : how reliable are they ? Dans *Annual Symposium Reliability and Maintainability, 2004 - RAMS*, 151–156.

Elsayed, E. (1996). *Reliability Engineering*. Numéro v. 1 de Reliability Engineering. Addison Wesley Longman.

FAA-HDBK-006A (2008). Federal Aviation Administration Handbook : Reliability, Maintainability, and Availability (RMA) Handbook. Récupéré de `https://www.quanterion.com/wp-content/uploads/2014/09/FAA-HDBK-006A.pdf`

Ford, D., Labelle, F., Popovici, F., Stokely, M., Truong, V.-A., Barroso, L., Grimes, C. et Quinlan, S. (2010). Availability in globally distributed storage systems. Dans *Symposium on Operating Systems Design and Implementation, 2010*.

Ghemawat, S., Gobioff, H. et Leung, S.-T. (2003). The google file system. Dans *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, SOSP '03, 29–43.

Goel, S. et Buyya, R. (2007). Data replication strategies in wide-area distributed systems. Dans *Enterprise service computing : from concept to deployment*, 211–241. IGI Global.

GR-2841-CORE (1994). Generic requirements for operations systems platform reliability. Récupéré de `http://telecom-info.telcordia.com/site-cgi/ido/docs.cgi?ID=SEARCH&DOCUMENT=GR-2841&`

Hamerly, G. et Elkan, C. (2001). Bayesian approaches to failure prediction for disk drives. Dans *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, 202–209., San Francisco, CA, USA. Morgan Kauf-

mann Publishers Inc. Récupéré de http://dl.acm.org/citation.cfm?id=645530.655825

Harnik, D., Naor, D. et Segall, I. (2009). Low power mode in cloud storage systems. Dans *2009 IEEE International Symposium on Parallel Distributed Processing*, 1–8. http://dx.doi.org/10.1109/IPDPS.2009.5161231

Hartigan, J. A. et Wong, M. A. (1979). Algorithm as 136 : A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*(1), 100–108.

Holland, J. H. (1975). *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press.

Hughes, G. F., Murray, J. F., Kreutz-Delgado, K. et Elkan, C. (2002). Improved Disk-Drive Failure Warnings. *IEEE Transactions on Reliability*, *51*(3), 350–357.

Janikow, C. Z. et Michalewicz, Z. (1991). An experimental comparison of binary and floating point representations in genetic algorithms. Dans *ICGA*, 31–36.

Kitano, H. (1990). Designing neural networks using genetic algorithms with graph generation system. *Complex Systems Journal*, *4*, 461–476.

Li, J., Ji, X., Jia, Y., Zhu, B., Wang, G., Li, Z. et Liu, X. (2014). Hard drive failure prediction using classification and regression trees. Dans *IEEE/IFIP Int. Conference on Dependable Systems and Networks.*

Li, W., Yang, Y., Chen, J. et Yuan, D. (2012). A cost-effective mechanism for cloud data reliability management based on proactive replica checking. Dans *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM Int. Symposium on.*

Li, W., Yang, Y. et Yuan, D. (2011). A novel cost-effective dynamic data replication strategy for reliability in cloud data centres. Dans *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, 496–502.

Liao, B., Yu, J., Sun, H. et Nian, M. (2012). A QoS-aware dynamic data replica deletion strategy for distributed storage systems under cloud computing environments. Dans *2012 Second International Conference on Cloud and Green Computing*, 219–225.

Lin, J. W., Chen, C. H. et Chang, J. M. (2013a). Qos-aware data replication for data-intensive applications in cloud computing systems. *IEEE Transactions*

on Cloud Computing, *1*(1), 101–115.

Lin, J. W., Chen, C. H. et Chang, J. M. (2013b). Qos-aware data replication for data-intensive applications in cloud computing systems. *IEEE Transactions on Cloud Computing*, *1*(1), 101–115.

Linden, G. (2006). Make data useful. Récupéré de `http://www.gduchamp.com/media/StanfordDataMining.2006-11-28.pdf`

Martello, S. et Toth, P. (1990). *Knapsack Problems : Algorithms and Computer Implementations*. New York, NY, USA : John Wiley & Sons, Inc.

Milligan, C. et Selkirk, S. (2002). Online storage virtualization : the key to managing the data explosion. Dans *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, 3052–3060.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. Cambridge, MA, USA : MIT Press.

Montana, D. J. et Davis, L. (1989). Training feedforward neural networks using genetic algorithms. Dans *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'89, 762–767., San Francisco, CA, USA.

Murray, J. F., Hughes, G. F. et Kreutz-Delgado, K. (2003). Hard drive failure prediction using non-parametric statistical methods. Dans *Proceedings of ICANN/ICONIP*.

Murray, J. F., Hughes, G. F. et Kreutz-Delgado, K. (2005). Machine learning methods for predicting failures in hard drives : A multiple-instance application. *J. Mach. Learn. Res.*, *6*, 783–816. Récupéré de `http://dl.acm.org/citation.cfm?id=1046920.1088699`

Pinheiro, E., Weber, W.-D. et Barroso, L. A. (2007). Failure trends in a large disk drive population. Dans *USENIX Conference on File and Storage Technologies*.

Rahman, R., Barker, K. et Alhajj, R. (2006). Replica placement design with static optimality and dynamic maintainability. Dans *Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE Int. Symposium on*, volume 1.

Sankar, S., Shaw, M., Vaid, K. et Gurumurthi, S. (2013). Datacenter scale evaluation of the impact of temperature on hard disk drive failures. *Trans. Storage*, *9*(2), 6 :1–6 :24.

Schroeder, B., Damouras, S. et Gill, P. (2010). Understanding latent sector errors and how to protect against them. *Trans. Storage*, *6*(3), 9 :1–9 :23.

Schroeder, B. et Gibson, G. (2010). A large-scale study of failures in high-performance computing systems. *Dependable and Secure Computing, IEEE Transactions on*, *7*(4), 337–350.

Schroeder, B. et Gibson, G. A. (2007). Disk failures in the real world : What does an mttf of 1, 000, 000 hours mean to you ? Dans *FAST*, volume 7, 1–16.

Schulze-Kremer, S. (1994). Genetic algorithms for protein tertiary structure prediction. Dans *Applications of Genetic Algorithms, IEE Colloquium on*, 6/1–6/5.

Shah, S. et Elerath, J. (2005). Reliability analysis of disk drive failure mechanisms. Dans *Reliability and Maintainability Symposium, 2005. Proceedings. Annual.*

Sharma, B., Chudnovsky, V., Hellerstein, J. L., Rifaat, R. et Das, C. R. (2011). Modeling and synthesizing task placement constraints in google compute clusters. Dans *Proceedings of the 2Nd ACM Symposium on Cloud Computing*, SOCC '11, 3 :1–3 :14.

Shvachko, K., Kuang, H., Radia, S. et Chansler, R. (2010). The hadoop distributed file system. Dans *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 1–10.

Somasundaram, G. et Shrivastava, A. (2009). *Information Storage and Management : Storing, Managing, and Protecting Digital Information.* EMC education services. Wiley.

Sombers, A. (2007). Let's get an availability benchmark. Récupéré de `http://www.availabilitydigest.com/private/0206/benchmarking.pdf`

Sun, D.-W., Chang, G.-R., Gao, S., Jin, L.-Z. et Wang, X.-W. (2012). Modeling a dynamic data replication strategy to increase system availability in cloud computing environments. *Journal of computer science and technology.*

Sun, H. et Han, J. J. (2001). Instantaneous availability and interval availability for systems with time-varying failure rate : stair-step approximation. Dans *Proceedings 2001 Pacific Rim International Symposium on Dependable Computing*, 371–374.

Syswerda, G. (1989). Uniform crossover in genetic algorithms. Dans *Proceedings of the 3rd International Conference on Genetic Algorithms*, 2–9., San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Récupéré de `http://dl.acm.org/citation.cfm?id=645512.657265`

Thompson, S. (2012). *Sampling.* CourseSmart. Wiley. Récupéré de `https://books.google.ca/books?id=-sFtXLIdDiIC`

Tollari, S. (2003). Algorithmes génétiques. Récupéré de http://sis.univ-tln.fr/~tollari/TER/AlgoGen1/node5.html

Vakilinia, S., Heidarpour, B. et Cheriet, M. (2016). Energy efficient resource allocation in cloud computing environments. *IEEE Access*, *4*, 8544–8557.

Vishwanath, K. V. et Nagappan, N. (2010). Characterizing cloud computing hardware reliability. Dans *Proceedings of the 1st ACM Symposium on Cloud Computing*, SoCC '10, 193–204., New York, NY, USA. ACM.

Wang, Y., Miao, Q. et Pecht, M. (2011). Health monitoring of hard disk drive based on mahalanobis distance. Dans *2011 Prognostics and System Health Managment Confernece*, 1–8. http://dx.doi.org/10.1109/PHM.2011.5939558

Wei, Q., Veeravalli, B., Gong, B., Zeng, L. et Feng, D. (2010). Cdrm : A cost-effective dynamic replication management scheme for cloud storage cluster. Dans *2010 IEEE International Conference on Cluster Computing*, 188–196.

Wright, A. H. *et al.* (1991). Genetic algorithms for real parameter optimization. *Foundations of genetic algorithms*, *1*, 205–218.

Xu, C., Wang, G., Liu, X., Guo, D. et Liu, T. Y. (2016). Health status assessment and failure prediction for hard drives with recurrent neural networks. *IEEE Transactions on Computers*, *65*(11), 3502–3508. http://dx.doi.org/10.1109/TC.2016.2538237

Yahoo (2015). S2 - yahoo ! statistical information regarding files and access pattern to files in one of yahoo's clusters. Accessed : 2016-03-24. Récupéré de http://webscope.sandbox.yahoo.com/catalog.php?datatype=s

Ye, Z., Zhou, X. et Bouguettaya, A. (2011). *Int. Conf. Database Systems for Advanced Applications*, chapitre Genetic Algorithm Based QoS-Aware Service Compositions in Cloud Computing.

Yue, M. (1991). A simple proof of the inequality ffd (l) 11/9 opt (l) + 1, l for the ffd bin-packing algorithm. *Acta Mathematicae Applicatae Sinica*, *7*(4), 321–331. http://dx.doi.org/10.1007/BF02009683. Récupéré de http://dx.doi.org/10.1007/BF02009683

Yue, M. et Zhang, L. (1995). A simple proof of the inequality mffd(l)71/60 opt(l) + 1,l for the mffd bin-packing algorithm. *Acta Mathematicae Applicatae Sinica*, *11*(3), 318–330. http://dx.doi.org/10.1007/BF02011198. Récupéré de http://dx.doi.org/10.1007/BF02011198

Zedlewski, J., Sobti, S., Garg, N., Zheng, F., Krishnamurthy, A. et Wang, R. (2003). Modeling hard-disk power consumption. Dans *Proceedings of the 2Nd*

USENIX Conference on File and Storage Technologies, FAST '03, 217–230., Berkeley, CA, USA. USENIX Association.

Zhang, Q., Hellerstein, J. et Boutaba, R. (2011). Characterizing task usage shapes in google compute clusters. Dans *Proceedings of the 5th Int. Workshop on Large Scale Distributed Systems and Middleware.*

Zhang, Q., Zhani, M. F., Boutaba, R. et Hellerstein, J. L. (2013). Harmony : Dynamic heterogeneity-aware resource provisioning in the cloud. Dans *2013 IEEE 33rd International Conference on Distributed Computing Systems*, 510–519.

Zhang, Q., Zhani, M. F., Zhang, S., Zhu, Q., Boutaba, R. et Hellerstein, J. L. (2012). Dynamic energy-aware capacity provisioning for cloud computing environments. Dans *Proceedings of the 9th International Conference on Autonomic Computing*, ICAC '12, 145–154., New York, NY, USA. ACM.

Zhani, M. F. (2011). *Prévision du trafic Internet : modèles et applications.* (Thèse de doctorat). Université du Québec à Montréal.

Zhao, H., Pan, M., Liu, X., Li, X. et Fang, Y. (2012). Optimal resource rental planning for elastic applications in cloud market. Dans *Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium*, IPDPS '12, 808–819., Washington, DC, USA. IEEE Computer Society.

Zhao, Y., Liu, X., Gan, S. et Zheng, W. (2010). Predicting disk failures with hmm- and hsmm-based approaches. Dans *Proceedings of the 10th Industrial Conference on Advances in Data Mining : Applications and Theoretical Aspects*, ICDM'10, 390–404., Berlin, Heidelberg. Springer-Verlag.

Zhu, B., Wang, G., Liu, X., Hu, D., Lin, S. et Ma, J. (2013). Proactive drive failure prediction for large scale storage systems. Dans *IEEE Symposium Mass Storage Systems and Technologies.*