Modèle Mathématique pour un Ordonnancement Dynamique et Adaptatif des Réplicas Kubernetes

MENRA W. ROMIAL

12 septembre 2025

1 Introduction

Contexte et Enjeu

L'avènement de Kubernetes comme standard incontournable pour l'orchestration de conteneurs a offert une élasticité remarquable pour le déploiement d'applications modernes. Cependant, cette agilité s'accompagne d'un défi de taille : la gestion de l'empreinte énergétique toujours croissante des infrastructures cloud. Face à la volatilité des sources d'énergie et à la pression croissante des régulations environnementales, les approches statiques de gestion de la puissance deviennent obsolètes. Il est désormais crucial d'aligner dynamiquement la consommation des ressources de calcul avec une enveloppe énergétique (budget de puissance) qui peut varier de manière imprévisible.

Problématique

Comment répartir équitablement et efficacement une contrainte de puissance fluctuante sur un ensemble hétérogène de microservices, tout en maintenant la stabilité et la réactivité du système ?

Proposition

Ce travail propose un modèle mathématique formel pour un controlleur Kubernetes qui répond à ce défi. Notre approche se distingue par sa conception en **boucle de contrôle adaptative** qui intègre plusieurs mécanismes :

- Une **machine à états** avec des fenêtres de stabilisation temporelles (W_{inc}, W_{dec}) et des seuils de tolérance (ϵ_{inc}, ϵ_{dec}) pour filtrer le bruit et ne déclencher des actions que sur des signaux durables.
- Une **répartition contextuelle et bidirectionnelle** de l'effort de scaling : la réduction de réplicas est proportionnelle à la consommation actuelle, ciblant ainsi les plus gros contributeurs, tandis que l'augmentation est proportionnelle à la capacité de croissance restante, favorisant une expansion équitable.
- Un **algorithme de redistribution** qui garantit le respect strict des contraintes de chaque application (nombre min et max de réplicas).

2 Formalisation du Modèle

2.1 Définitions Fondamentales

Définition 2.1 (Ensembles)**.**
- $\mathcal{D}$: L'ensemble de tous les déploiements sujets au scaling.
- $\mathcal{G}$: L'ensemble de tous les groupes de déploiements.
- $\mathcal{D}_g \subset \mathcal{D}$: L'ensemble des déploiements appartenant au groupe $g \in \mathcal{G}$.

Définition 2.2 (Paramètres par Déploiement)**.** Pour chaque déploiement $d \in \mathcal{D}$:
- $C_d(t) \in \mathbb{N}^*$: Le nombre de réplicas **actuels** à l'instant t.
- $Min_d \in \mathbb{N}^*$: Le nombre **minimum** de réplicas requis.
- $Max_d \in \mathbb{N}^*$: Le nombre **maximum** de réplicas configuré.
- $w_d \in \mathbb{R}^+$: Le **poids de répartition** au sein d'un groupe.

Le système maintient à tout instant la contrainte $Min_d \leq C_d(t) \leq Max_d$.

Définition 2.3 (Paramètres et État du Contrôleur)**.**
- $P(t) \in \mathbb{R}^+$: La puissance disponible mesurée à l'instant t.
- $W_{inc}, W_{dec} \in \mathbb{R}^+$: Les durées des fenêtres de stabilisation.
- $\epsilon_{inc}, \epsilon_{dec} \in [0, 1]$: Les marges de tolérance pour la stabilisation.
- $\delta \in [0, 1]$: Le seuil de changement relatif minimal (deadband).

— $P_{ref} \in \mathbb{R}^+$: La dernière puissance de référence pour laquelle une action a été effectuée.
— mode $\in$ {STABLE, PENDING_INCREASE, PENDING_DECREASE} : L'état de la machine à états.
— $t_{trigger}, P_{trigger} \in \mathbb{R}^+$: L'instant et la puissance lors du déclenchement d'un changement.

Définition 2.4 (Variable de Décision)**.**
— $x_d \in \mathbb{Z}$: Le nombre de réplicas à ajouter ($x_d > 0$) ou à retirer ($x_d < 0$).

3 Modèle Algorithmique : La Boucle de Contrôle

3.1 Phase 1 : Gestion des États et Déclenchement (Le Contrôleur)

Cette phase agit comme un filtre pour décider si une action de scaling est justifiée.

3.1.1 Cas : `mode = STABLE`

1. Mesurer la puissance actuelle $P(t)$.
2. Calculer la variation relative : $\Delta P_{rel} = (P(t) - P_{ref})/P_{ref}$.
3. **Si** $|\Delta P_{rel}| < \delta$, le changement est insignifiant. Ne rien faire.
4. **Si** $\Delta P_{rel} \geq \delta$ (augmentation significative) :
   — Changer mode $\rightarrow$ PENDING_INCREASE.
   — Enregistrer $t_{trigger} \leftarrow t$ et $P_{trigger} \leftarrow P(t)$.
5. **Si** $\Delta P_{rel} \leq -\delta$ (réduction significative) :
   — Changer mode $\rightarrow$ PENDING_DECREASE.
   — Enregistrer $t_{trigger} \leftarrow t$ et $P_{trigger} \leftarrow P(t)$.

3.1.2 Cas : `mode = PENDING_INCREASE`

1. Mesurer la puissance actuelle $P(t)$.
2. **Si** $(t - t_{trigger}) < W_{inc}$, la fenêtre de stabilisation n'est pas terminée. Attendre.
3. **Sinon** (la fenêtre est terminée) :
   — Vérifier la stabilité : si $|P(t) - P_{trigger}|/P_{trigger} \leq \epsilon_{inc}$, le signal est considéré comme stable.
      — **Déclencher le calcul de scaling (Phase 2)** avec la variation de puissance $\Delta P = P(t) - P_{ref}$.
      — Après l'action, mettre à jour $P_{ref} \leftarrow P(t)$ et revenir à mode $\leftarrow$ STABLE.
   — Sinon, le signal a trop varié. L'augmentation potentielle est annulée.
      — Mettre à jour $P_{ref} \leftarrow P(t)$ et revenir à mode $\leftarrow$ STABLE.

La logique pour le cas `PENDING_DECREASE` est parfaitement symétrique, en utilisant W_{dec} et ϵ_{dec}.

3.2 Phase 2 : Calcul et Répartition du Scaling

Cette phase est déclenchée lorsque les conditions de stabilisation sont remplies.

3.2.1 Calcul de l'objectif global de réplicas

L'objectif de variation est calculé de manière à ce que la variation relative du nombre de réplicas soit proportionnelle à la variation relative de la puissance. Soit $C_{total}(t) = \sum_{d \in \mathcal{D}} C_d(t)$ le nombre total de réplicas actuels.

$$\Delta_{\text{total}} = \text{round}\left(C_{total}(t) \cdot \frac{P(t) - P_{ref}}{P_{ref}} \right) \in \mathbb{Z} \tag{1}$$

Si $\Delta_{\text{total}} = 0$, aucune action n'est nécessaire.

3.2.2 Répartition de l'objectif entre entités

La répartition de $|\Delta_{\text{total}}|$ est contextuelle et symétrique.

Cas A : Scale-Down ($\Delta_{\text{total}} < 0$) L'effort de réduction est réparti au prorata de la **consommation actuelle** de chaque entité e. Cette approche est juste car elle cible les contributeurs principaux à la consommation présente. La consommation d'une entité est $C_e(t) = \sum_{d \in e} C_d(t)$. La cible de réduction brute est :

$$\Delta_e^{\text{brut}} = |\Delta_{\text{total}}| \cdot \frac{C_e(t)}{\sum_k C_k(t)} \tag{2}$$

Cas B : Scale-Up ($\Delta_{\text{total}} > 0$) L'ajout de réplicas est réparti au prorata de la **capacité d'accueil restante** de chaque entité e. La capacité restante d'une entité est $A_e = \sum_{d \in e}(Max_d - C_d(t))$. La cible d'augmentation brute est :

$$\Delta_e^{\text{brut}} = \Delta_{\text{total}} \cdot \frac{A_e}{\sum_k A_k} \tag{3}$$

Dans les deux cas, la **méthode du plus grand reste** est utilisée pour obtenir des cibles entières Δ_e dont la somme est exactement $|\Delta_{\text{total}}|$.

3.2.3 Calcul de la variation voulue par déploiement (x_d^{voulue})

À cette étape, nous traduisons les cibles par entité (Δ_d, Δ_g) en variations voulues pour chaque déploiement individuel.

Cas des déploiements indépendants Pour un déploiement $d \in \mathcal{D}_I$, la variation voulue correspond simplement à la cible calculée pour lui, affectée du signe de la variation globale.

$$x_d^{\text{voulue}} = \text{sign}(\Delta_{\text{total}}) \cdot \Delta_d \tag{4}$$

Cas des déploiements groupés Pour un groupe $g \in \mathcal{G}$, nous répartissons sa cible (qui est une magnitude) Δ_g entre ses déploiements membres $d \in \mathcal{D}_g$. Cette répartition se fait au prorata des poids de répartition w_d.

— Somme des poids du groupe g : $W_g = \sum_{k \in \mathcal{D}_g} w_k$

La variation brute pour chaque $d \in \mathcal{D}_g$ est :

$$\delta_d^{\text{brut}} = \Delta_g \cdot \frac{w_d}{W_g} \tag{5}$$

Nous appliquons la méthode du plus grand reste sur les valeurs δ_d^{brut} pour obtenir des parts entières δ_d, garantissant que $\sum_{d \in \mathcal{D}_g} \delta_d = \Delta_g$. La variation voulue finale est alors obtenue en appliquant le signe de la variation globale :

$$x_d^{\text{voulue}} = \text{sign}(\Delta_{\text{total}}) \cdot \delta_d \tag{6}$$

3.2.4 Ajustement final sous contraintes et redistribution

Cette étape garantit que $Min_d \leq C_d(t) + x_d \leq Max_d$ pour tous les déploiements.

Cas A : Scale-Down ($x_d^{\text{voulue}} < 0$)

1. **Réduction max possible** : $x_d^{\text{max\_possible}} = -(C_d(t) - N_d)$.
2. **Réduction initiale** : $x_d^{(0)} = \max(x_d^{\text{voulue}}, x_d^{\text{max\_possible}})$.
3. **Déficit** : Déficit $= \sum_d (x_d^{\text{voulue}} - x_d^{(0)})$, qui est ≤ 0.
4. **Redistribution itérative** : Tant que Déficit < 0, on identifie le déploiement "donneur" d^* avec la plus grande marge $(C_d(t) - Min_d)$ et on lui applique une réduction supplémentaire de 1.

Cas B : Scale-Up ($x_d^{\text{voulue}} > 0$)

1. **Augmentation max possible** : $x_d^{\text{max\_possible}} = Max_d - C_d(t)$.
2. **Augmentation initiale** : $x_d^{(0)} = \min(x_d^{\text{voulue}}, x_d^{\text{max\_possible}})$.
3. **Surplus** : Surplus $= \sum_d (x_d^{\text{voulue}} - x_d^{(0)})$, qui est ≥ 0.
4. **Redistribution itérative** : Tant que Surplus > 0, on identifie le déploiement "receveur" d^* avec la plus grande capacité restante $(Max_d - C_d(t))$ et on lui applique une augmentation supplémentaire de 1.

4 Synthèse du Résultat Final

Au terme de l'algorithme, nous obtenons pour chaque déploiement une valeur finale $x_d^{\text{final}} \in \mathbb{Z}$. Le nouvel état désiré du cluster est :

$$C'_d = C_d(t) + x_d^{\text{final}} \tag{7}$$

Ce modèle garantit par construction que $Min_d \leq C'_d \leq Max_d$ pour tout $d \in \mathcal{D}$.

5 Conclusion et Perspectives

Le modèle formalisé dans ce document constitue une base complète pour un ordonnanceur Kubernetes conscient de l'énergie et adapté aux conditions réelles.

— **Robustesse :** La gestion d'état et les fenêtres de stabilisation le rendent résilient au bruit et aux signaux volatils.
— **Adaptabilité :** Le calcul de l'objectif de scaling basé sur la variation relative s'ajuste dynamiquement à l'échelle du cluster.
— **Équité contextuelle :** La répartition de la charge (réduction ou augmentation) est basée sur l'état actuel du système, ce qui assure des décisions plus justes et plus efficaces.

Les perspectives d'évolution incluent l'intégration de métriques de Qualité de Service (QoS) comme condition supplémentaire avant de valider une action de scaling.