Résilience et auto-réparation de processus de décisions multi-agents

Pierre Rust[1,2]   Gauthier Picard[1]   Fano Ramparany[2]

[1]MINES Saint-Étienne, CNRS
Lab Hubert Curien UMR 5516

[2]Orange Labs

Prise de décision distribuée dans des SMA dynamiques

Décision

- Problème d'optimisation sous contraintes
- Décisions $\equiv$ variables (COP)

Distribution

Dynamique

Prise de décision distribuée dans des SMA dynamiques

Décision

- Problème d'optimisation sous contraintes
- Décisions $\equiv$ variables (COP)

Distribution

- Multi-agent
- Optimisation distribuée (DCOP)
- Distribution **efficace** des décisions

Dynamique

Prise de décision distribuée dans des SMA dynamiques

Décision

- Problème d'optimisation sous contraintes
- Décisions $\equiv$ variables (COP)

Distribution

- Multi-agent
- Optimisation distribuée (DCOP)
- Distribution **efficace** des décisions

Dynamique

- Agents arrivants/partants
- Les décisions doivent être préservées
- Les décisions doivent être migrées

Distribution des décisions

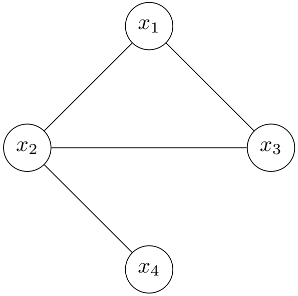
Prise de décision distribuée $\Rightarrow$ DCOP : $\langle \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{C}, \mu \rangle$

- $\mathcal{A} = \{a_1, \ldots, a_{|A|}\}$ un ensemble d'agents ;
- $\mathcal{X} = \{x_1, \ldots, x_n\}$ un ensemble de variables ;
- $\mathcal{D} = \{\mathcal{D}_{x_1}, \ldots, \mathcal{D}_{x_n}\}$ les domaines des variables x_i ;
- $\mathcal{C} = \{c_1, \ldots, c_m\}$ un ensemble de contraintes souples
- μ une fonction associant chaque variable à un agent $\Rightarrow$ La **distribution**.

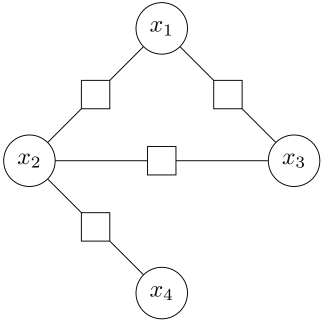
Une *solution* au DCOP est une affectation de valeur à toutes les variables qui minimise la somme totale des coûts $\sum_i c_i$.

Distribution des décisions

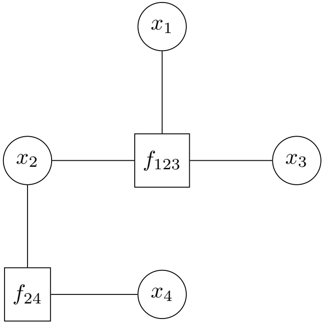
- Algorithmes pour les DCOP, optimaux ou pas
- Plusieurs représentations graphiques
- Nœuds = calculs
- Distribuer les calculs sur les agents



(a) Graphe de contraintes



(b) Graphe de facteurs binaires

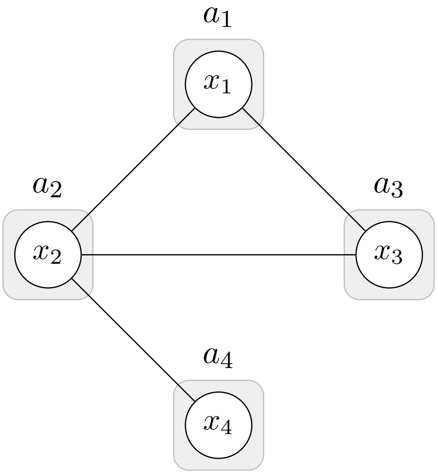


(c) Graphes de facteurs n-aires

Distribuer les calculs

Calcul

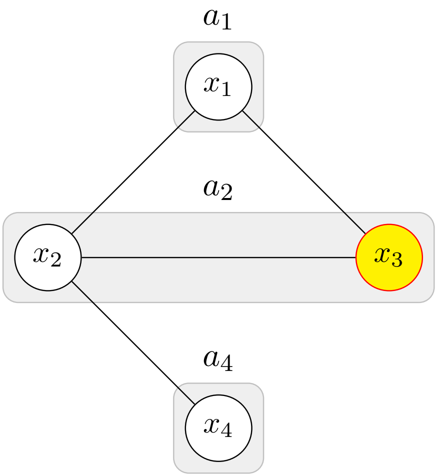
- **appartiennent** à un agent :
  lien "naturel",
  spécifique au problème



Distribuer les calculs

Calcul

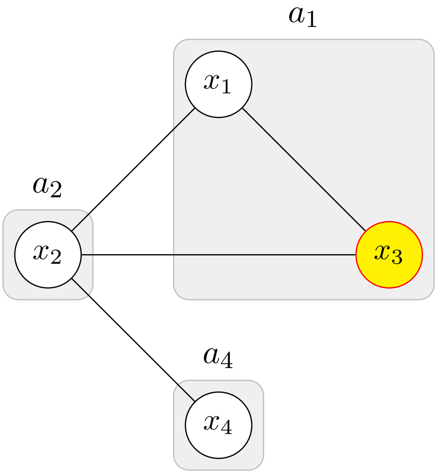
- **appartiennent** à un agent
- décision **partagée** :
  artefact de modélisation, avec aucun lien évident avec un agent (e.g. emploi du temps distribué)



Distribuer les calculs

Calcul

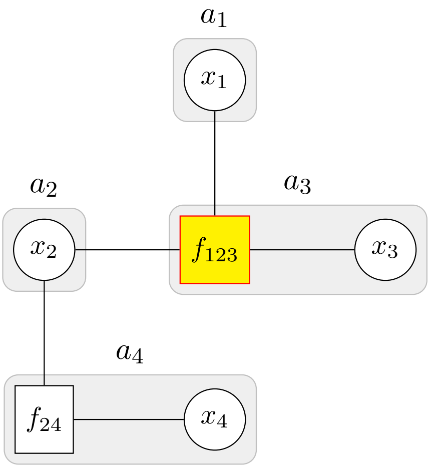
- **appartiennent** à un agent
- décision **partagée** :
  artefact de modélisation, avec aucun lien évident avec un agent (e.g. emploi du temps distribué)



Distribuer les calculs

Calcul

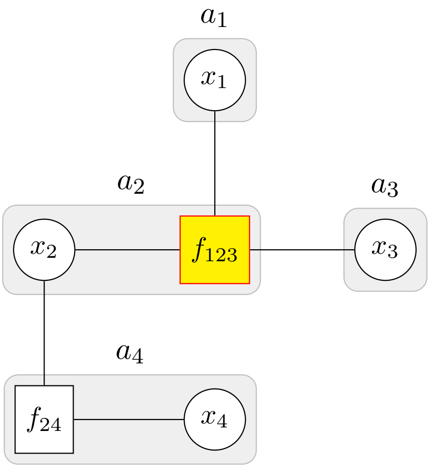
- **appartiennent** à un agent
- décision **partagée**
- **facteur**, dans un graphe de facteurs :
  ne représente pas une variable de décision



Distribuer les calculs

Calcul

- **appartiennent** à un agent
- décision **partagée**
- **facteur**, dans un graphe de facteurs :
  ne représente pas une variable de décision



Distribuer les calculs *(cont.)*

La distribution impacte les caractéristiques du système

- vitesse d'exécution
- charge de communication
- coût d'hébergement / préférences

Distribution optimale

- dépendante du problème
- problème d'optimisation : trouver la meilleur distribution suivant vos critères
- déterminer la distribution optimale $\equiv$ partitionnement de graphe
  NP-difficile en général [Boulle, 2004]

Définition d'une distribution optimale

Définition générique, basée sur

- La **capacité** des agents :
  $\rightarrow \mathbf{cap}(a_m)$ & $\mathbf{weight}(x_i)$
- la charge de **communication** :
  avec des coût de communication entre agents différents
  $\rightarrow \mathbf{com}(x_i, y_j, a_m, a_n)$
- les préférences $\rightarrow$ **coûts d'hébergement** :
  peut être utilisé pour modéliser des préférences, des coûts opérationnels, etc.
  $\rightarrow \mathbf{host}(a_m, x_i)$

Définition d'une distribution optimale

Définition générique

- Respecter les capacités limites des agents & empreinte de calculs

$$\forall a_m \in \mathbf{A}, \quad \sum_{x_i \in D} \mathbf{weight}(x_i) \cdot x_i^m \leq \mathbf{cap}(a_m) \tag{1}$$

Définition d'une distribution optimale

Définition générique

- Respecter les capacités limites des agents & empreinte de calculs

$$\forall a_m \in \mathbf{A}, \quad \sum_{x_i \in D} \mathbf{weight}(x_i) \cdot x_i^m \leq \mathbf{cap}(a_m) \tag{1}$$

- Minimiser la charge communicationnelle :
  avec des coût de communication entre agents différents

$$\underset{x_i^m}{\mathbf{minimiser}} \quad \sum_{(i,j) \in D} \sum_{(m,n) \in \mathbf{A}^2} \mathbf{com}(i,j,m,n) \cdot \alpha_{ij}^{mn} \tag{2}$$

Définition d'une distribution optimale

Définition générique

- Respecter les capacités limites des agents & empreinte de calculs

$$\forall a_m \in \mathbf{A}, \quad \sum_{x_i \in D} \mathbf{weight}(x_i) \cdot x_i^m \leq \mathbf{cap}(a_m) \tag{1}$$

- Minimiser la charge communicationnelle :
  avec des coût de communication entre agents différents

$$\underset{x_i^m}{\mathbf{minimiser}} \quad \sum_{(i,j) \in D} \sum_{(m,n) \in \mathbf{A}^2} \mathbf{com}(i,j,m,n) \cdot \alpha_{ij}^{mn} \tag{2}$$

- Minimiser les coûts d'hébergement :
  peut être utiliser pour modéliser des préférences, des coûts opérationnels, etc.

$$\underset{x_i^m}{\mathbf{minimiser}} \sum_{(x_i, a_m) \in X \times \mathbf{A}} x_i^m \cdot \mathbf{host}(a_m, x_i) \tag{3}$$

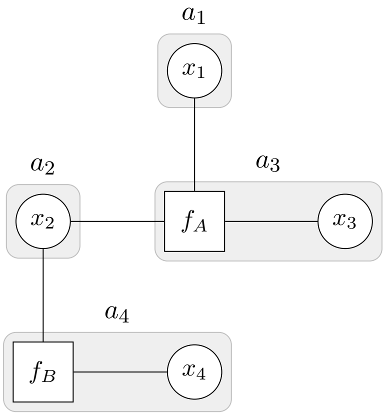
Distribuer les calculs *(cont.)*

Distribution optimale

- NP-difficile, mais peut être résolue avec un *branch-and-cut*
  Les solveur de programmes linéaire sont assez bons pour ça
- Utile pour *bootstraper* le système
- Mais, seulement possible pour de petites instances
- Lorsque ce n'est pas résolvable, donne une métrique de qualité des distributions

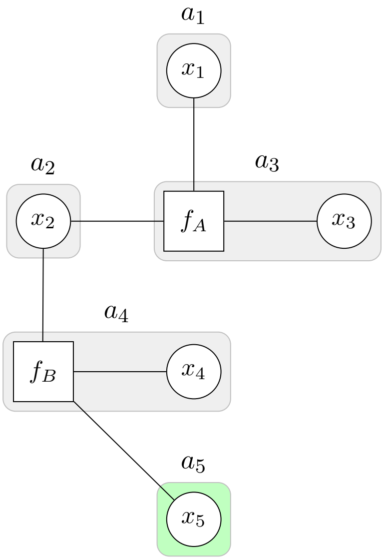
Systèmes ouverts et dynamiques

- Les calculs sont distribués sur les agents :
  que se passe-t-il si le système change ?



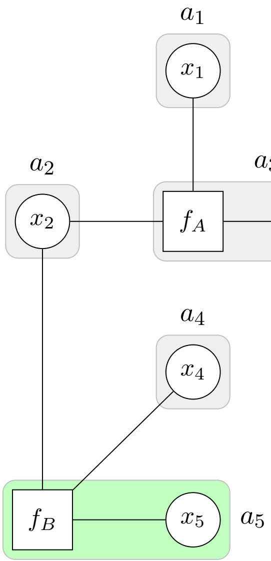
Systèmes ouverts et dynamiques

- De nouveaux agents peuvent **rejoindre** le système
  - Utiliser de l'effort/capacité de calcul supplémentaire ?
  - Migrer des calculs ?



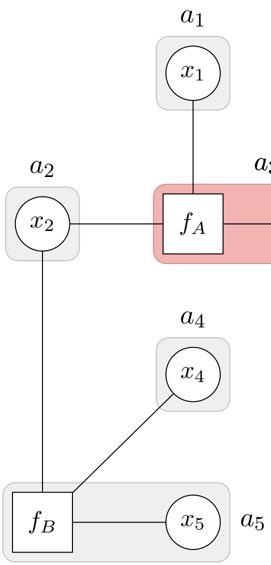
Systèmes ouverts et dynamiques

- De nouveaux agents peuvent **rejoindre** le système
  - Utiliser de l'effort/capacité de calcul supplémentaire ?
  - Migrer des calculs ?



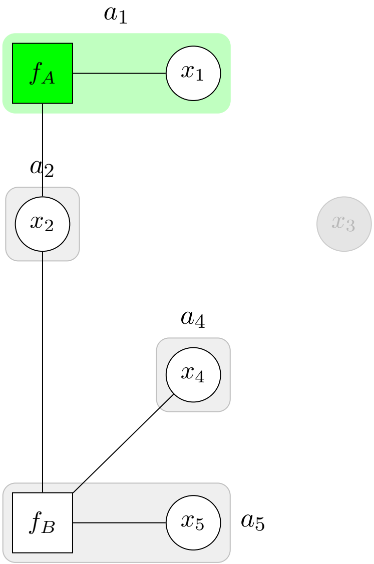
Systèmes ouverts et dynamiques

- De nouveaux agents peuvent **rejoindre** le système
  - Utiliser de l'effort/capacité de calcul supplémentaire ?
  - Migrer des calculs ?
- Des agents peuvent **quitter** le système à tout moment
  - Comment assurer que le système marche encore de manière nominale ?
  - Migrer les calculs aux agents restants



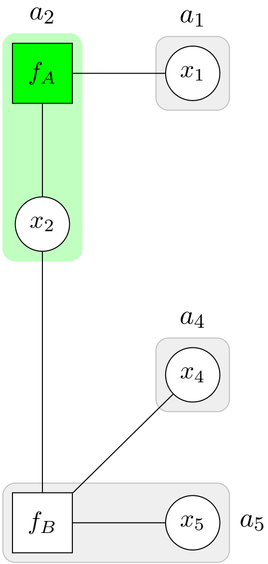
Systèmes ouverts et dynamiques

- De nouveaux agents peuvent **rejoindre** le système
  - Utiliser de l'effort/capacité de calcul supplémentaire ?
  - Migrer des calculs ?
- Des agents peuvent **quitter** le système à tout moment
  - Comment assurer que le système marche encore de manière nominale ?
  - Migrer les calculs aux agents restants



Systèmes ouverts et dynamiques

- De nouveaux agents peuvent **rejoindre** le système
  - Utiliser de l'effort/capacité de calcul supplémentaire ?
  - Migrer des calculs ?
- Des agents peuvent **quitter** le système à tout moment
  - Comment assurer que le système marche encore de manière nominale ?
  - Migrer les calculs aux agents restants



k-résilience

Définition (k-résilience)

Etant donnés les agents $\mathbf{A}$, les calculs $\mathbf{X}$, et une distribution μ, un système est k-**résilient** si pour tout $F \subset \mathbf{A}, |F| \leq k$, une nouvelle distribution $\mu' : X \to \mathbf{A} \setminus F$ existe

Mise en œuvre

- Rendre disponible les définition des calculs : **réplication**
- Migrer les calculs « orphelins » : **sélection** du candidat

Réplication pour la k-résilience

Placement des réplicas

- Répliquer les calculs sur k agents
- Respecter les capacités des agents
- Optimiser les coûts de communication et d'hébergement

Réplication optimale (?)

- Très grand espace de recherche $\equiv$ problème du sac à dos multiple quadratique (QMKP) [Saraç et Sipahioglu, 2014], NP-difficile
- Pas de définition claire de l'optimalité

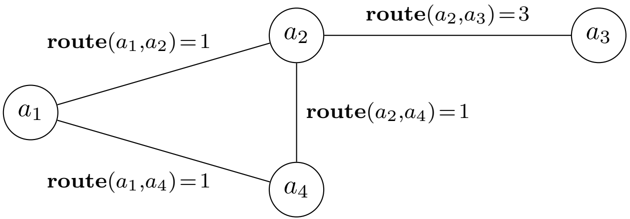
Réplication pour la k-résilience (cont.)

Approche heuristique : *Distributed Replica Placement Method* (DRPM)

- Exploiter le graphe de calculs : coûts de communication
- Ajouter des nœuds supplémentaires pour les coûts d'hébergement
- Utiliser l'algorithme *Iterative Lengthening / Uniform Cost Search* sur le graphe obtenu
- Implantation distribuée
- Initiée par chaque agent, pour chacun de ses calculs à répliquer

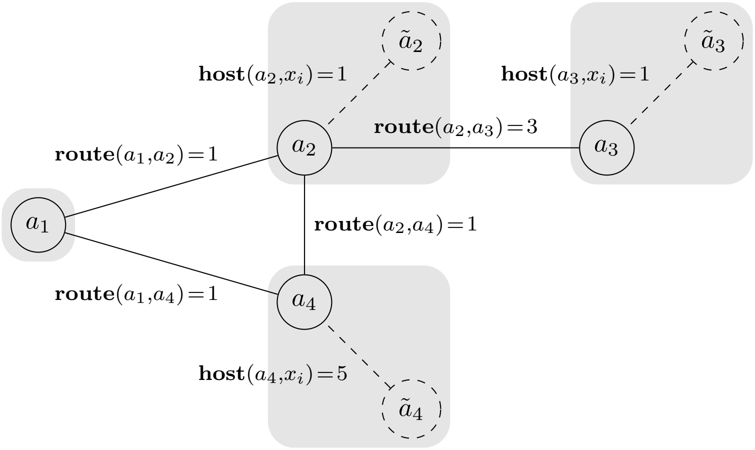
Distributed Replica Placement Method (DRPM)

Placement de 2 réplicas pour x_i, hébergé sur a_1



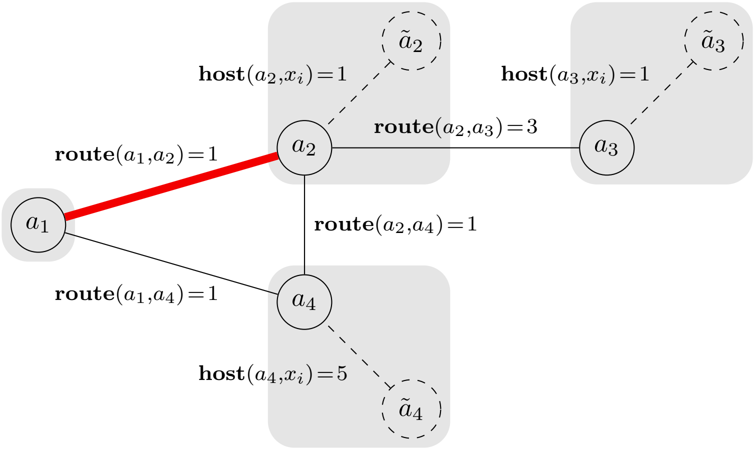
Distributed Replica Placement Method (DRPM)

Placement de 2 réplicas pour x_i, hébergé sur a_1



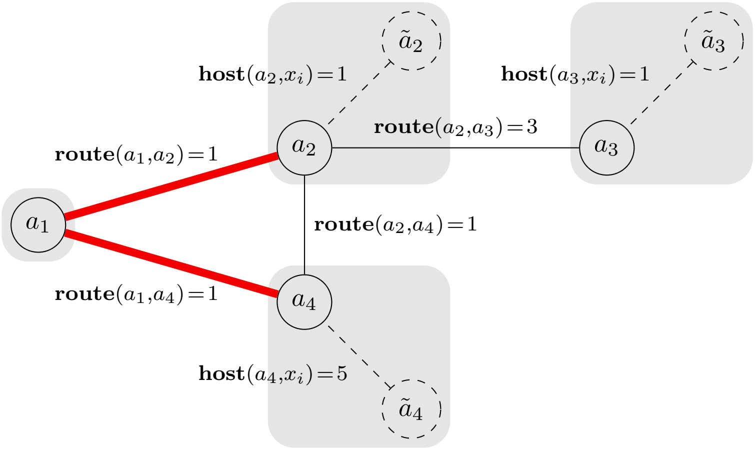
Distributed Replica Placement Method (DRPM)

Budget de 1



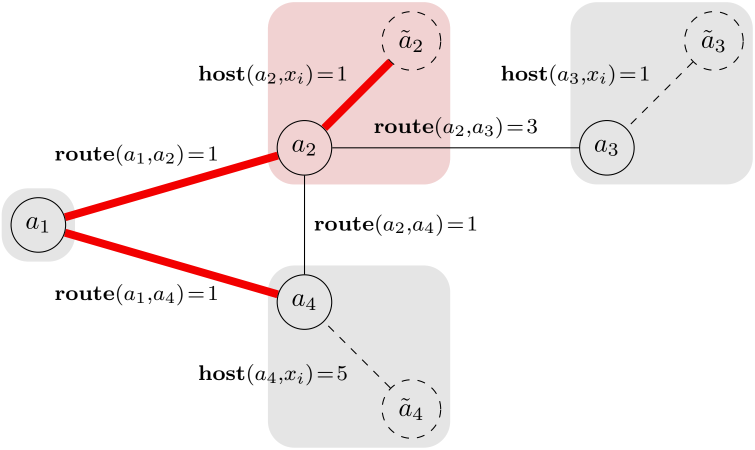
Distributed Replica Placement Method (DRPM)

Budget de 1



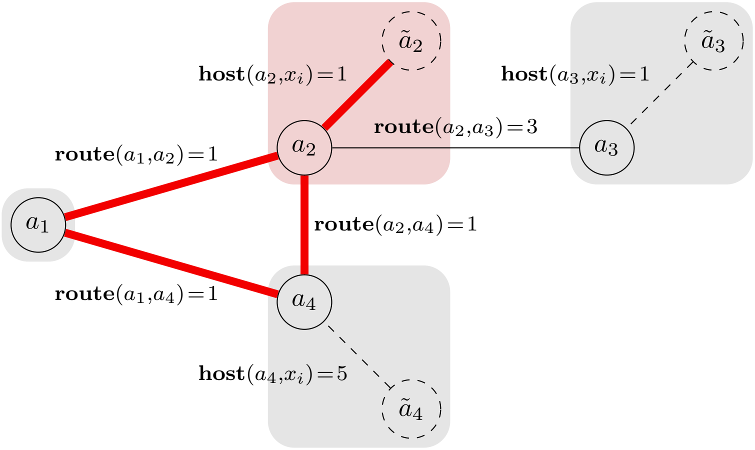
Distributed Replica Placement Method (DRPM)

Budget de 2 : placement d'un replica sur a_2



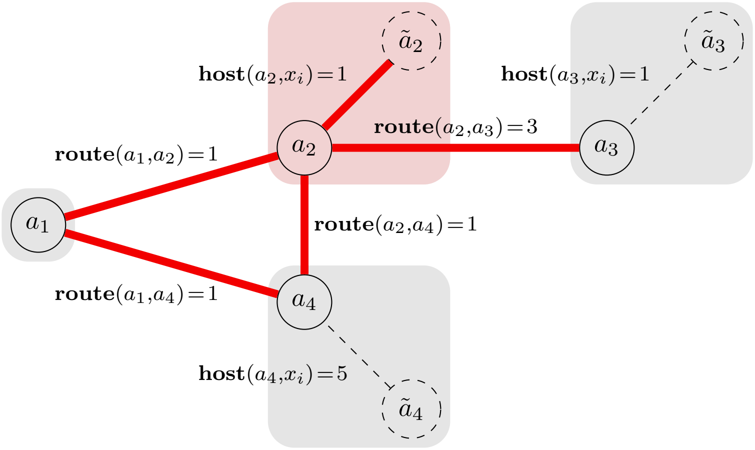
Distributed Replica Placement Method (DRPM)

Budget de 2 : placement d'un replica sur a_2



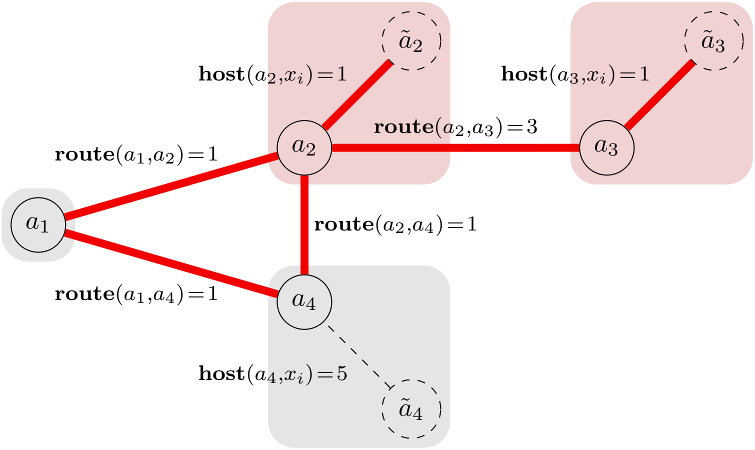
Distributed Replica Placement Method (DRPM)

Budget de 4



Distributed Replica Placement Method (DRPM)

Budget de 5 : deuxième replica sur a_3



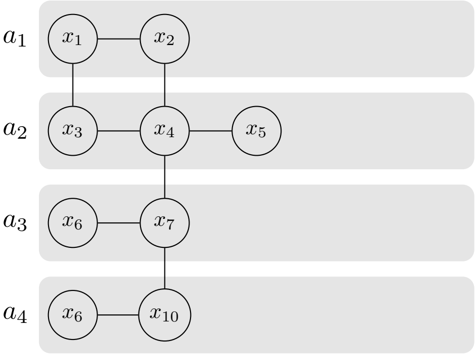
Réparation distribuée : sélection des candidats

Réparer en migrant les calculs

- Calculs orphelins X_c : hébergés sur l'agent disparu
- Agents candidats A_c : agents possédants des réplicas des calculs orphelins ($\leq k$ pour chaque calcul)
- Sélectionner exactement un agent candidat pour chaque calcul orphelin
- Respecter les capacités des agents
- Sélectionner les candidats qui minimisent les coûts de communication et d'hébergement

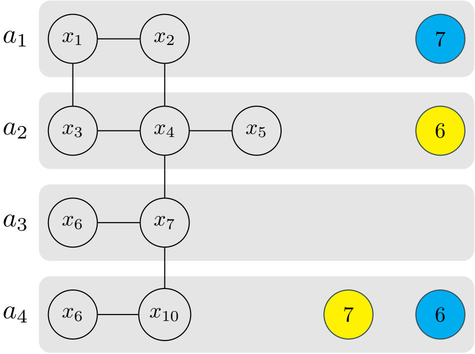
Similaire au problème de distribution initial mais sur un sous-graphe très restreint du graphe initial.

Réparation distribuée : sélection des candidats (*cont.*)



DCOP avec 9 calculs distribués sur 4 agents

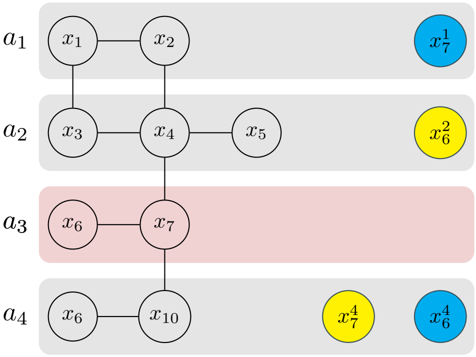
Réparation distribuée : sélection des candidats (*cont.*)



DCOP avec 9 calculs distribués sur 4 agents

- Réplicas de x_6 hébergés sur a_2, a_4
- Réplicas de x_7 hébergés sur a_1, a_4

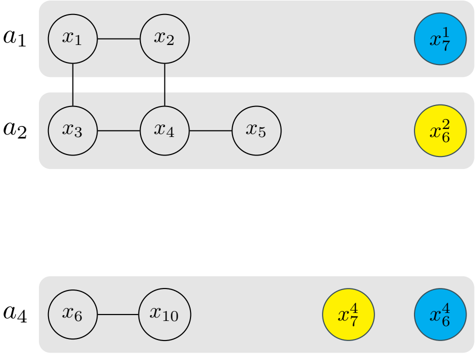
Réparation distribuée : sélection des candidats (*cont.*)



a_3 quitte le système

- x_6 et x_7 doivent être replacés
- Agents candidats :
  x_6 : { a_2, a_4 }
  x_7 : { a_1, a_4 }
- Variable binaire pour chaque réplica : x_i^m

Réparation distribuée : sélection des candidats (*cont.*)

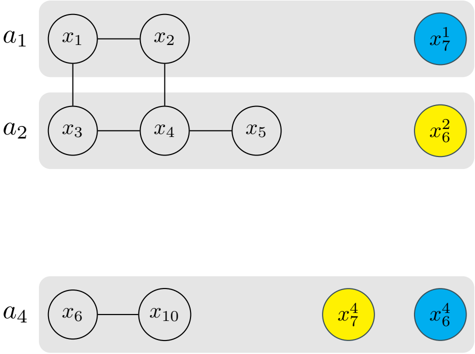


Modélisation de la sélection comme un problème d'optimisation

Tous les calculs orphelins doivent être hébergés :

$$\sum_{a_m \in A_c^i} x_i^m = 1 \qquad (4)$$

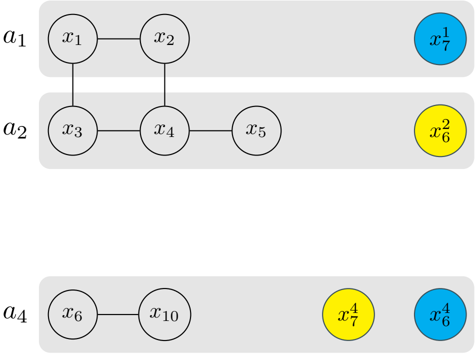
Réparation distribuée : sélection des candidats (*cont.*)



Contraintes de capacité

$$\sum_{x_i \in X_c^m} \mathbf{weight}(x_i) \cdot x_i^m + \sum_{x_j \in \mu^{-1}(a_m) \setminus X_c} \mathbf{weight}(x_j) \leq \mathbf{cap}(a_m) \quad (4)$$

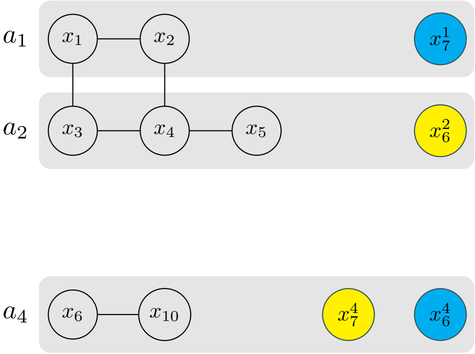
Réparation distribuée : sélection des candidats (*cont.*)



Minimisation des coûts d'hébergement

$$\sum_{x_i \in X_c^m} \mathbf{host}(a_m, x_i) \cdot x_i^m \tag{4}$$

Réparation distribuée : sélection des candidats (*cont.*)



Minimisation des coûts de communication

$$\sum_{(x_i,x_j)\in X_c^m \times N_i \setminus X_c} x_i^m \cdot \mathbf{com}(i,j,m,\mu^{-1}(x_j)) + \sum_{(x_i,x_j)\in X_c^m \times N_i \cap X_c} x_i^m \cdot \sum_{a_n \in A_c^j} x_j^n \cdot \mathbf{com}(i,j,m,n)) \quad (4)$$

Résoudre le problème de la sélection

Problème de décision optimale

- Modélisé comme un DCOP
- Nous utilisons un DCOP pour réparer la distribution du DCOP original!
  - DCOP original : variables = décisions pour notre problème
  - DCOP de réparation : variables = sélection des candidats

Résolution

- Algorithme MGM-2 [Maheswaran et al., 2004]
- Rapide, léger, monotone
- Bon comportement avec les contraintes souples et dures
- Pas de problème de distribution dans ce cas

Expérimentations

3 types de DCOP

- coloration de graphes aléatoires avec densité $p = 0.3$
- coloration de graphes *scale free* [Barabási, 1999]
- modèles d'Ising

2 types d'infrastructures

- uniforme
- dépendante du problème

2 processus de décision distribuées

- A-DSA [Zhang et al., 2005]
- Max-Sum [Farinelli et al., 2008]

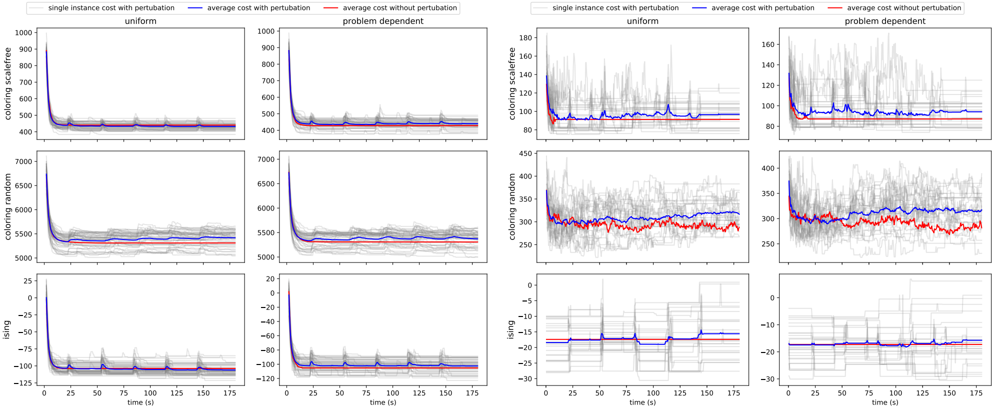
Scénarios de perturbation

- toutes les 30 secondes, 3 agents disparaissent
- MGM-2 pour la réparation

20 instances générées, 5 exécutions par instance
Résolution avec ou sans perturbations
Implantation avec pyDCOP : `https://github.com/Orange-OpenSource/pyDcop`

Résultats expérimentaux

Coût moyen des solutions



(a) A-DSA

(b) Max-Sum

FIGURE – Coût des solutions de Max-Sum (a) et A-DSA (b) en cours d'exécution, avec (bleu) et sans perturbations (rouge), sur des infrastructures uniformes (gauche) et dépendantes du problème (droite), et sur coloration *scale free* (haut), coloration aléatoire (milieu), et Ising (bas)

Résultats expérimentaux

A-DSA - Coloring Scalefree - Problem dependant

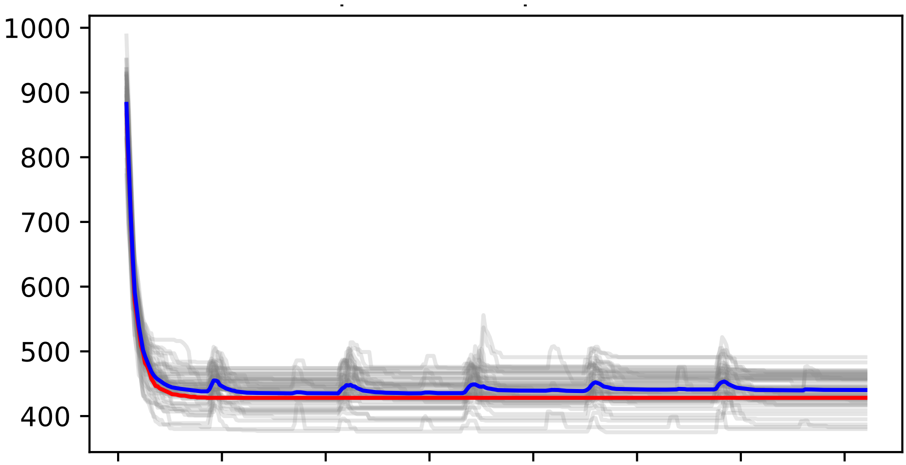


FIGURE – A-DSA - Scalefree - Problem dependant

Résultats expérimentaux

A-DSA - Ising - Uniform

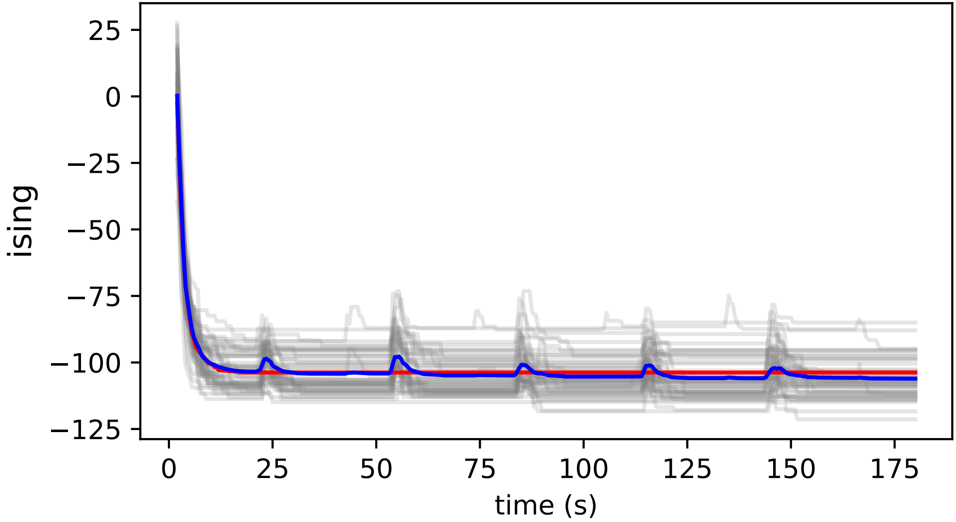
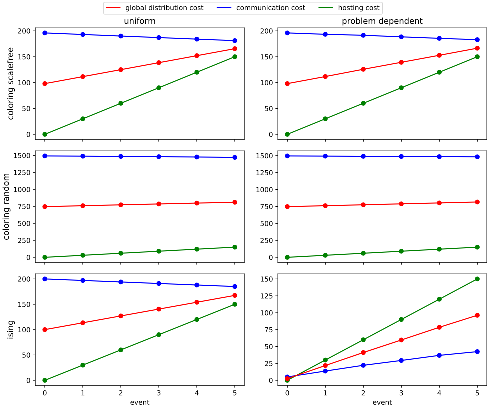


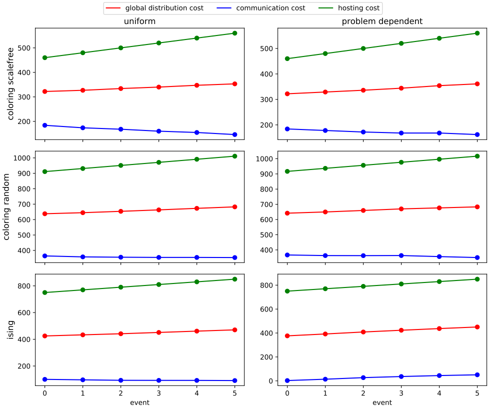
FIGURE – A-DSA - Ising - Uniform

Résultats expérimentaux

Coût moyen des distributions



(a) A-DSA



(b) Max-Sum

FIGURE – Coût de la distribution des graphes de calculs pour A-DSA et Max-Sum, après chaque événement, sur des infrastructures uniformes (gauche) et dépendantes du problème (droite), et sur coloration *scale free* (haut), coloration aléatoire (milieu), et Ising (bas)

Conclusion

Résumé

- Définition d'une distribution optimale des calculs/décisions sur un ensemble d'agents
- Algorithme distribué de réplication des calculs
- Méthode de réparation distribuée, basée sur un DCOP

Travaux futurs

- Relâcher les contraintes sur l'algorithme DCOP utilisé pour le problème original
- Tester avec d'autres algorithmes (seulement Max-Sum et DSA)
- Plus d'expérimentations avec d'autres domaines applicatifs
  - Autres types de problèmes (*meeting scheduling*, *target tracking*, etc.)
  - Graphe de calculs non DCOP (e.g. VNF)

Résilience et auto-réparation de processus de décisions multi-agents

Pierre Rust[1,2] Gauthier Picard[1] Fano Ramparany[2]

[1]MINES Saint-Étienne, CNRS
Lab Hubert Curien UMR 5516

[2]Orange Labs

Références

Barabási, A. (15 oct. 1999). “Emergence of Scaling in Random Networks”. In : *Science* 286.5439, p. 509-512. ISSN : 00368075, 10959203. DOI : 10.1126/science.286.5439.509. URL : http://www.sciencemag.org/cgi/doi/10.1126/science.286.5439.509 (visité le 15/11/2018).

Boulle, M. (2004). “Compact Mathematical Formulation for Graph Partitioning”. In : *Optimization and Engineering* 5.3, p. 315-333. ISSN : 1573-2924. DOI : 10.1023/B:OPTE.0000038889.84284.c7. URL : http://dx.doi.org/10.1023/B:OPTE.0000038889.84284.c7.

Farinelli, A., A. Rogers, A. Petcu et N. R. Jennings (2008). “Decentralised Coordination of Low-power Embedded Devices Using the Max-sum Algorithm”. In : *International Conference on Autonomous Agents and Multiagent Systems (AAMAS’08)*, p. 639-646. ISBN : 978-0-9817381-1-6. URL : http://dl.acm.org/citation.cfm?id=1402298.1402313.

Maheswaran, R.T., J.P. Pearce et M. Tambe (2004). “Distributed Algorithms for DCOP : A Graphical-Game-Based Approach”. In : *Proc. of the 17th International Conference on Parallel and Distributed Computing Systems (PDCS)*, p. 432-439.

Saraç, T. et A. Sipahioglu (2014). “Generalized quadratic multiple knapsack problem and two solution approaches”. In : *Computers & Operations Research* 43.Supplement C, p. 78 -89. ISSN : 0305-0548. DOI : https://doi.org/10.1016/j.cor.2013.08.018. URL : http://www.sciencedirect.com/science/article/pii/S0305054813002244.

Zhang, W., G. Wang, Z. Xing et L. Wittenburg (jan. 2005). “Distributed stochastic search and distributed breakout : properties, comparison and applications to constraint optimization problems in sensor networks”. In : *Artificial Intelligence* 161.1, p. 55-87. ISSN : 00043702. DOI : 10.1016/j.artint.2004.10.004. URL : http://linkinghub.elsevier.com/retrieve/pii/S0004370204001481 (visité le 29/08/2018).

Résultats expérimentaux

A-DSA - Coloring Random - Uniform

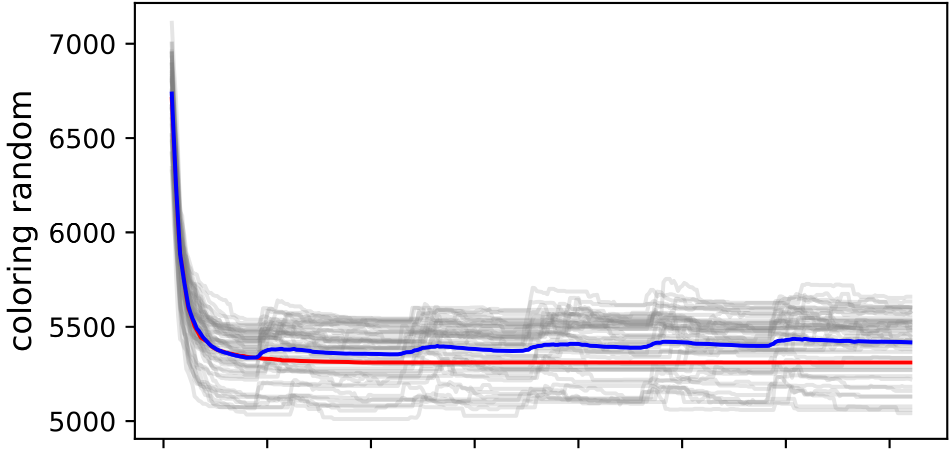


FIGURE – A-DSA - Coloring Random - Uniform

Résultats expérimentaux

A-DSA - Coloring Random - Dependant

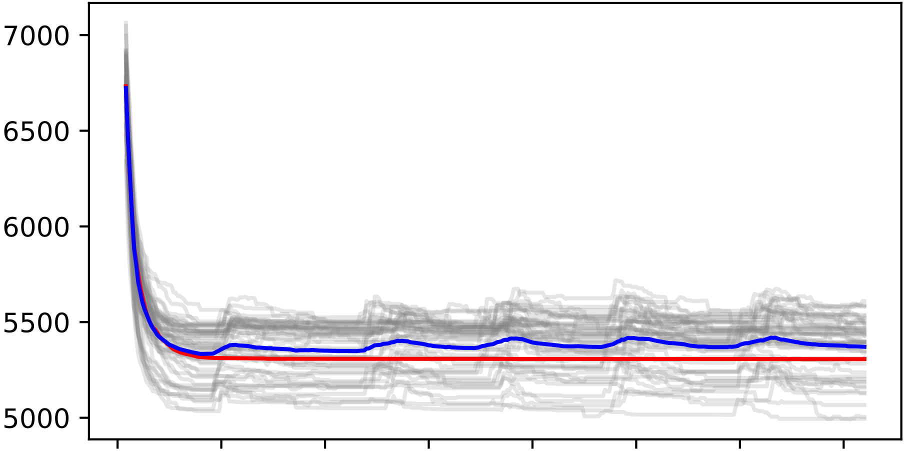


FIGURE – A-DSA - Coloring Random - Dependant

Résultats expérimentaux

MaxSum - Ising - Uniform

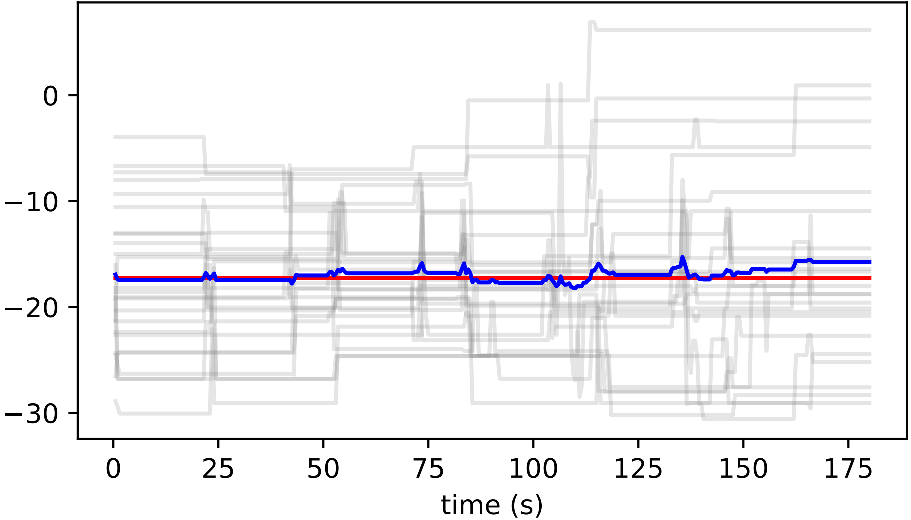


FIGURE – MaxSum - Ising - Uniform

Résultats expérimentaux

MaxSum - Coloring Random - Uniform

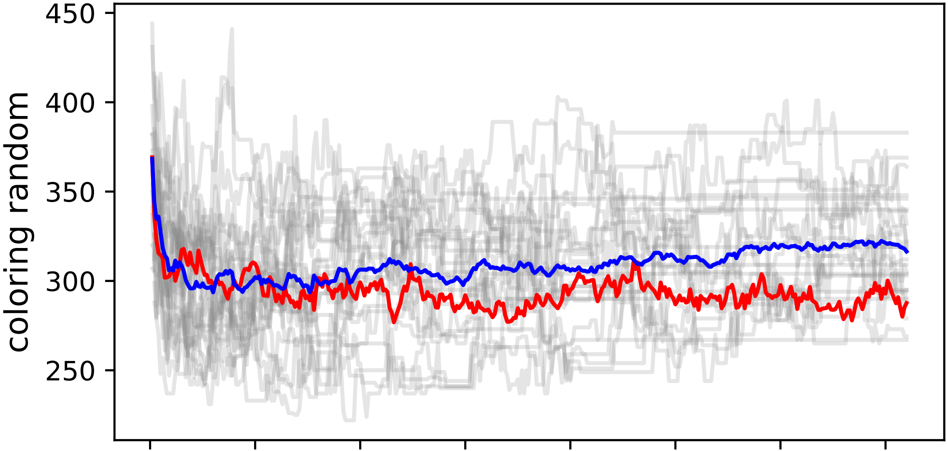


FIGURE – MaxSum - Coloring Random - Uniform