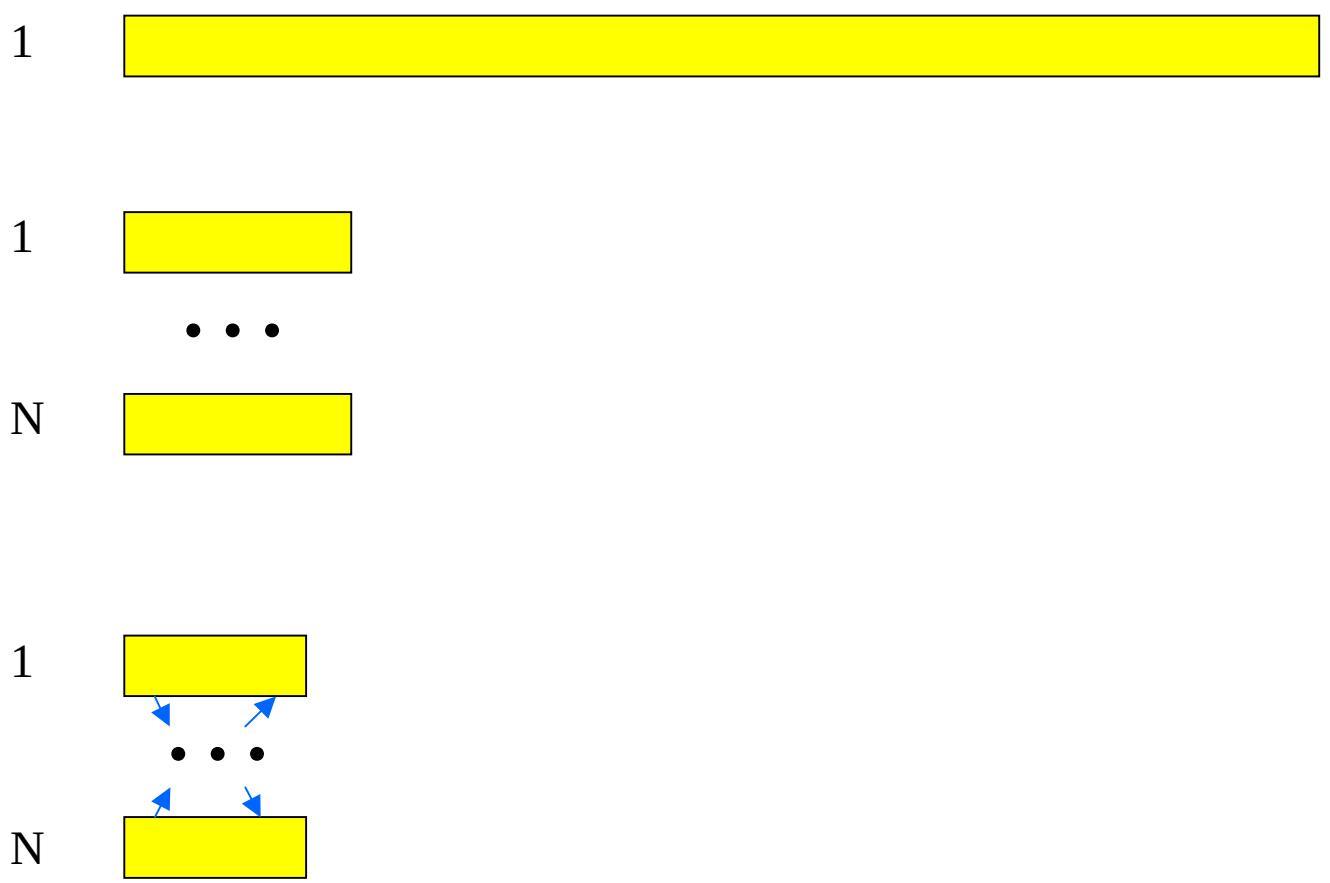
Programmation par Contraintes Distribuée

Kim BENNI
Université Paris VI

Philippe MORIGNOT
AXLOG Ingéniérie

Intuition



1

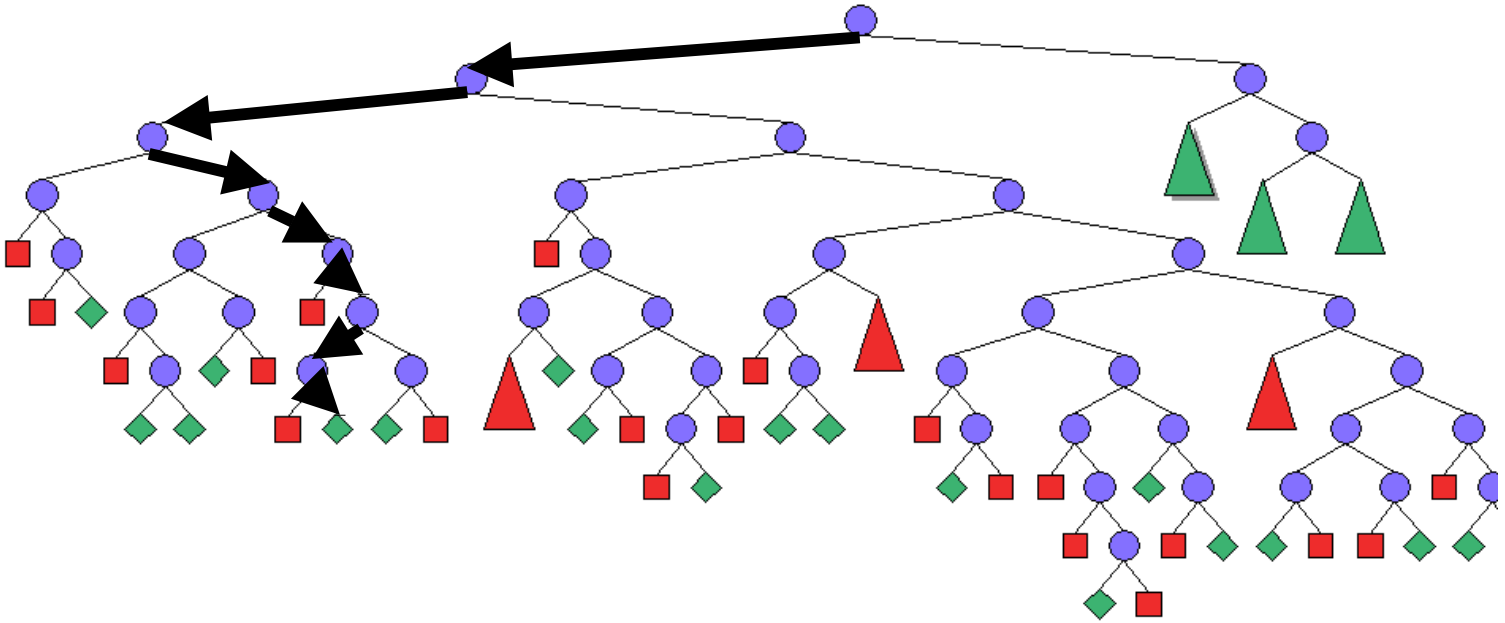
$\frac{1}{N}$

$< \frac{1}{N} ?$

Contexte

- Programmation par contraintes.
- Formellement, soit :
  - X_i des variables
  - D_i des domaines discrets (e.g., à valeurs entières)
  - C_j des relations n-aires entre les variables X_i
- Problème :
  - trouver une valeur dans chaque D_i pour chaque variable X_i, qui satisfasse toutes les contraintes C_j.
  - … avec une fonction à minimiser.
- Exemple : $x \in [1,10], \quad y \in [1,10], \quad x < y \quad \text{et} \quad \min(x+y)$

Approche : estimateur [Knuth 75]



- Convergence vers la valeur exacte (Monte Carlo)
- Complexité polynomiale
- Valeurs estimées : profondeur, nombre de solutions, nombre de nœuds

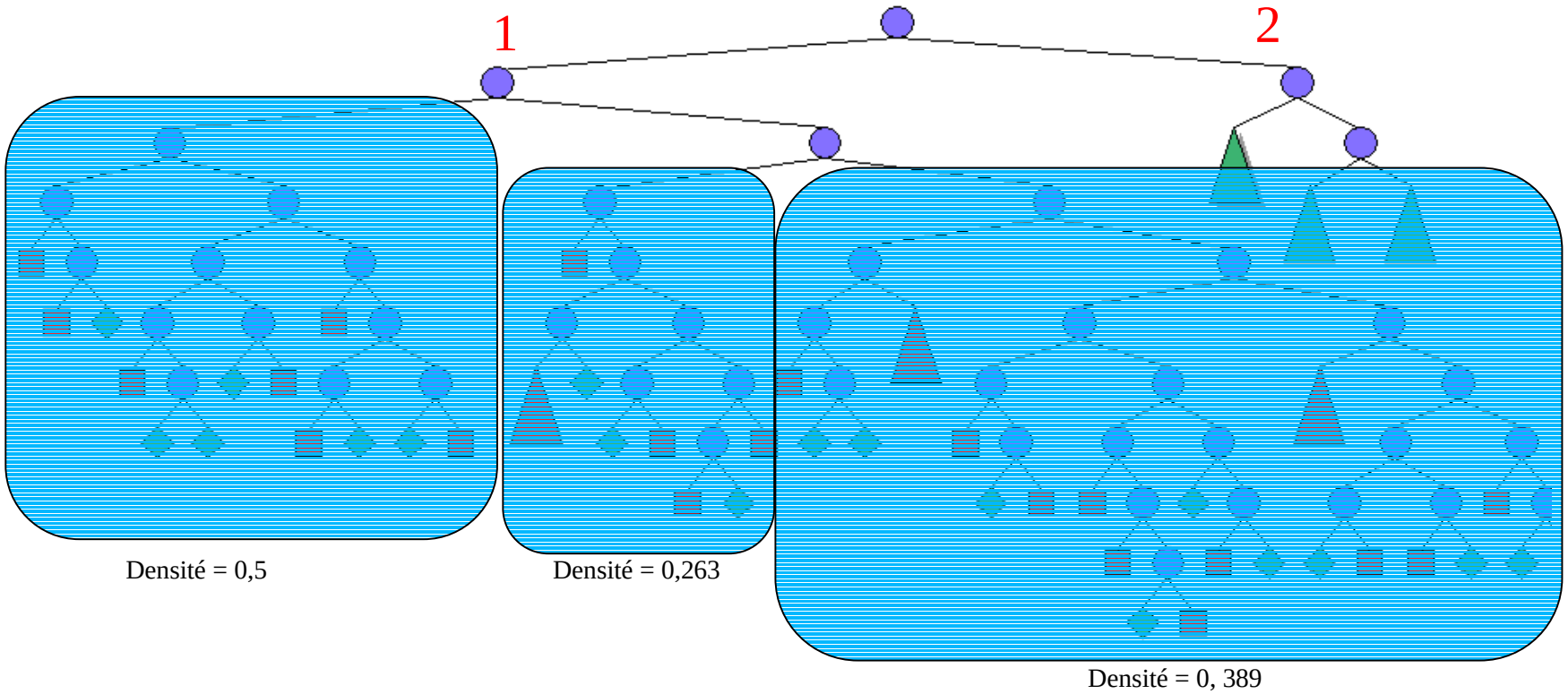
Approche : découpage (1/2)

*52 sous-noeuds
*10 niveaux
*18 solutions

Densité de solution = 0.346

*330 sous-noeuds
*16 niveaux
*74 solutions

Densité de solution = 0.224



Approche : découpage (2/2)

- Problème : rechercher une liste de nœuds.
- Contraintes :
  - Tous les nœuds de la liste doivent être distincts.
  - L'ensemble des nœuds terminaux doit être inclus dans l'ensemble des nœuds des sous-arbres déterminés par les nœuds de la liste.
  - Si un nœud est dans la liste, aucun de ses descendants n'y est.
- Minimiser l'écart de taille, maximiser l'écart de densité de solutions --- entre les sous-arbres.

➢ Programmation par contraintes.

Approche : communication

- Asynchrone.
- Quand un agent trouve une solution, il envoie son coût (et sa solution) aux autres agents.
  - Élagage des solutions de coût supérieur pour les autres agents.
  - Tolérance aux pannes (mort d'un agent).

Implémentation

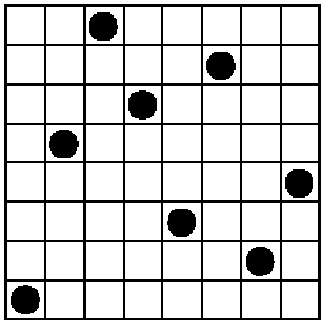
- Langage Oz
- Environnement Mozart
- ~ 4000 lignes de code

```
fun {Queens N}
   proc {$ Row}
      L1N  = {List.number 1 N 1}
      LM1N = {List.number ~1 ~N ~1}
   in
      Row = {FD.tuple queens N 1#N}
      {FD.distinct Row}
      {FD.distinctOffset Row L1N}
      {FD.distinctOffset Row LM1N}
      {FD.distribute ff Row}
   end
  end
end
```

Exemple de code source OZ

Méthodologie : exemples

- N-Reines :



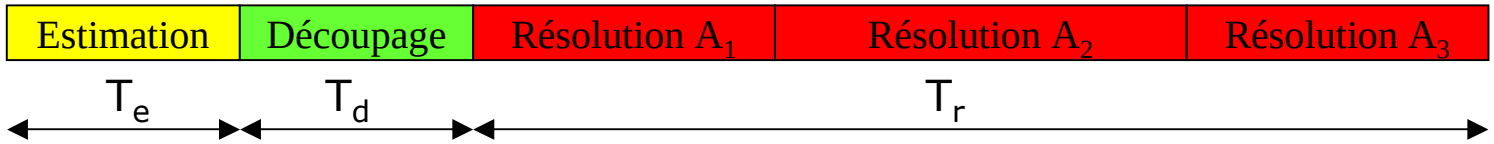
- Fugue de J.–S. Bach :

Voix	Premier mouvement	Second mouvement	Troisième mouvement	Quatrième mouvement
Soprano	Blanc	Blanc	Sujet	Contre-sujet
Alto	Blanc	Réponse	Contre-sujet	Blanc
Ténor	Sujet	Contre-sujet	Blanc	Blanc
Basse	Blanc	Blanc	Blanc	Réponse

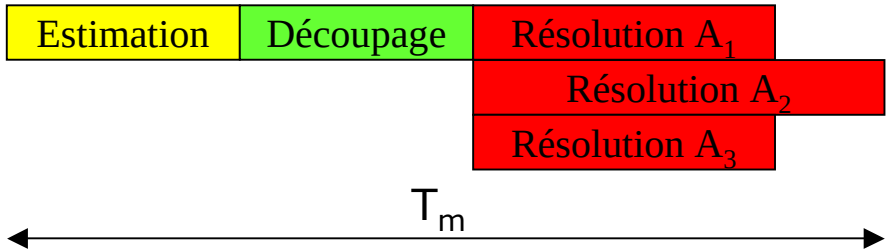
  - La taille du contre-sujet est égale au sujet.
  - L'intervalle est d'un octave au maximum par rapport à la tonalité du sujet.
  - Les notes du contre-sujet sont exclusivement des noires, des blanches ou des croches.
  - Il ne doit jamais y avoir de notes à l'unisson entre le sujet et le contre-sujet.
  - Il ne doit jamais y avoir de notes formant une quinte entre le sujet et le contre-sujet.
  - Le contre-sujet ne peut comporter que des notes faisant partie de la gamme utilisée par le sujet.
  - Une dominante du contre-sujet ne peut pas être précédée par une note qui est 7, 9 ou 11 demi-tons plus basse.
  - La réponse est constituée des notes du sujet majoré de 7 demi-tons.

Méthodologie : mesures

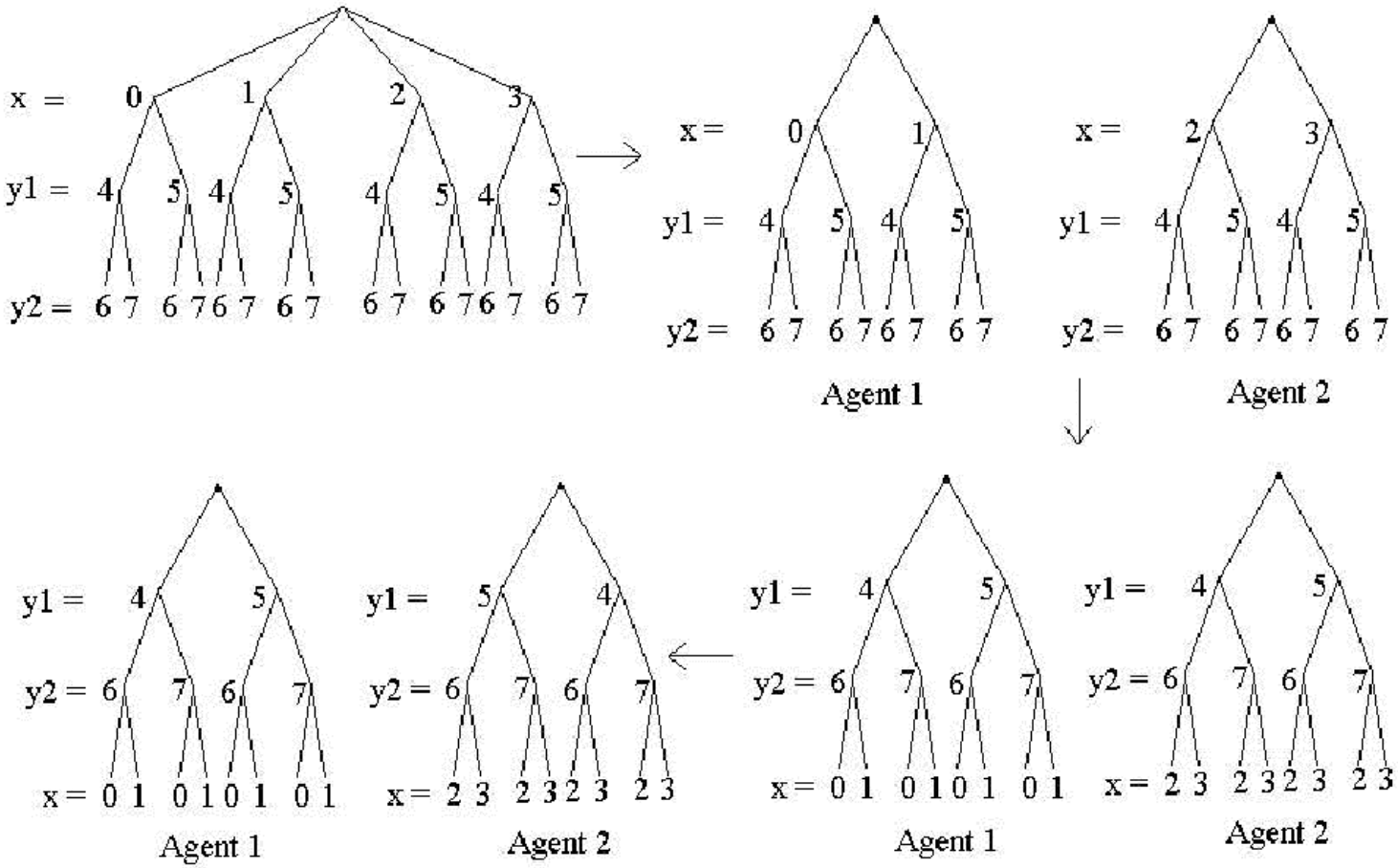
- 1 processus :
  - Branch & Bound



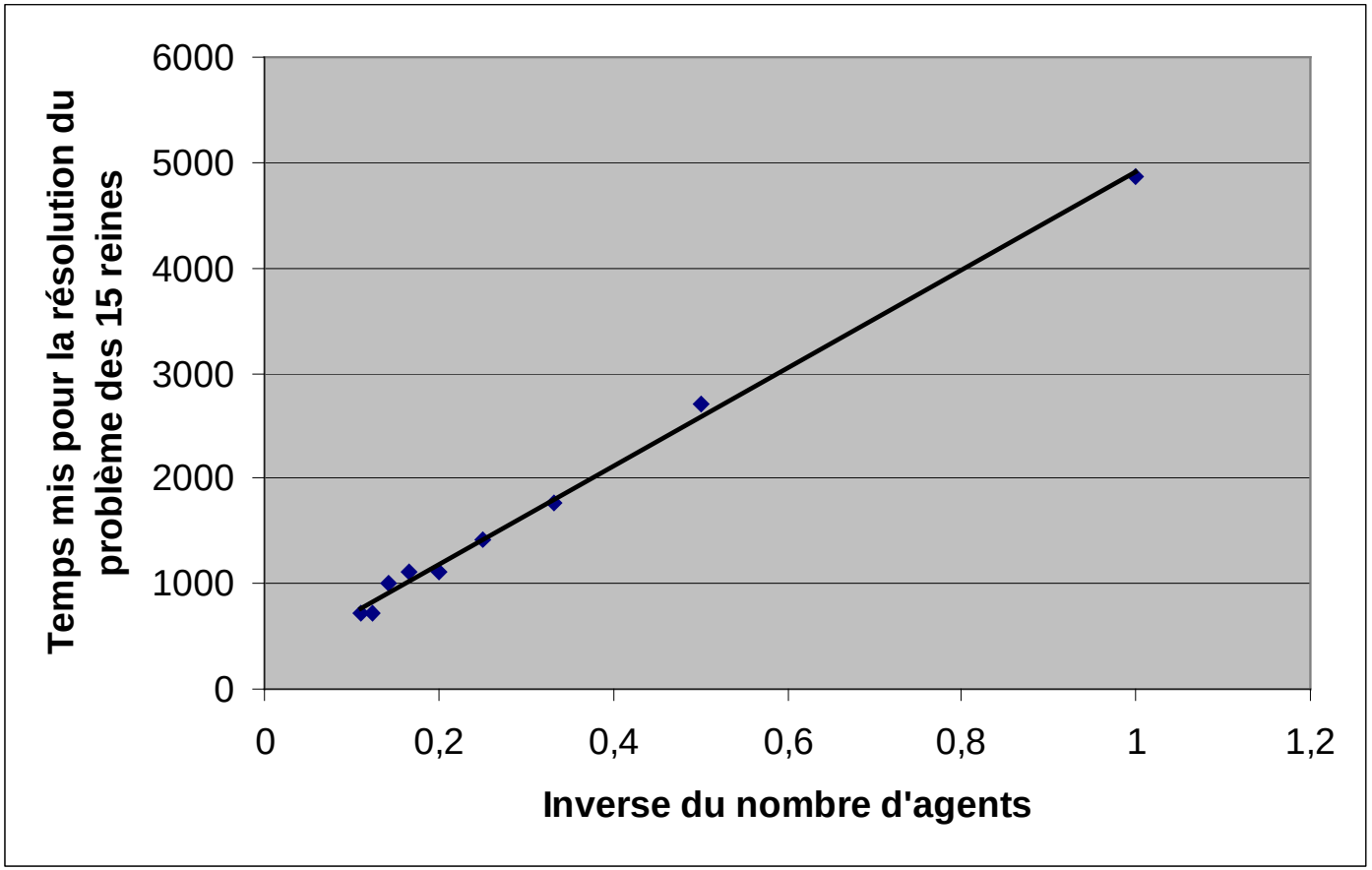
- Multi-processus :
  - Découpage « naïf »



Découpage « naïf » (1/2) [Gorge & Morignot 05]

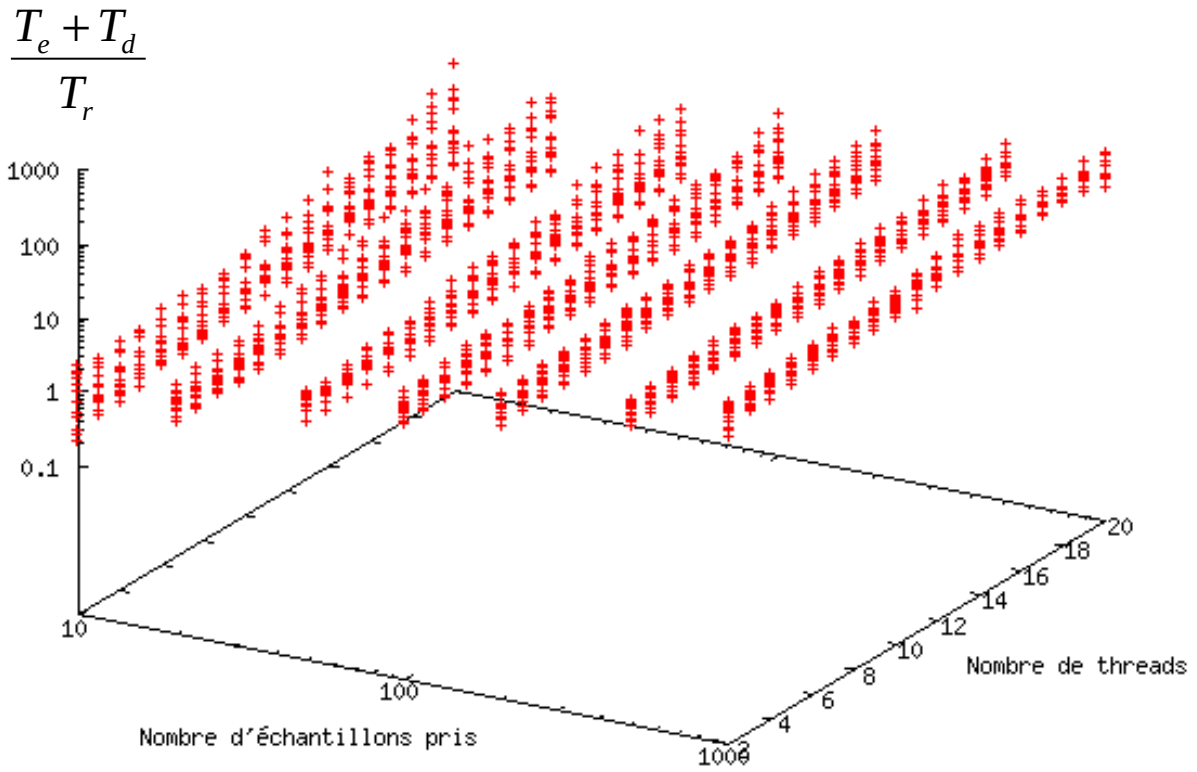


Découpage « naïf » (2/2)



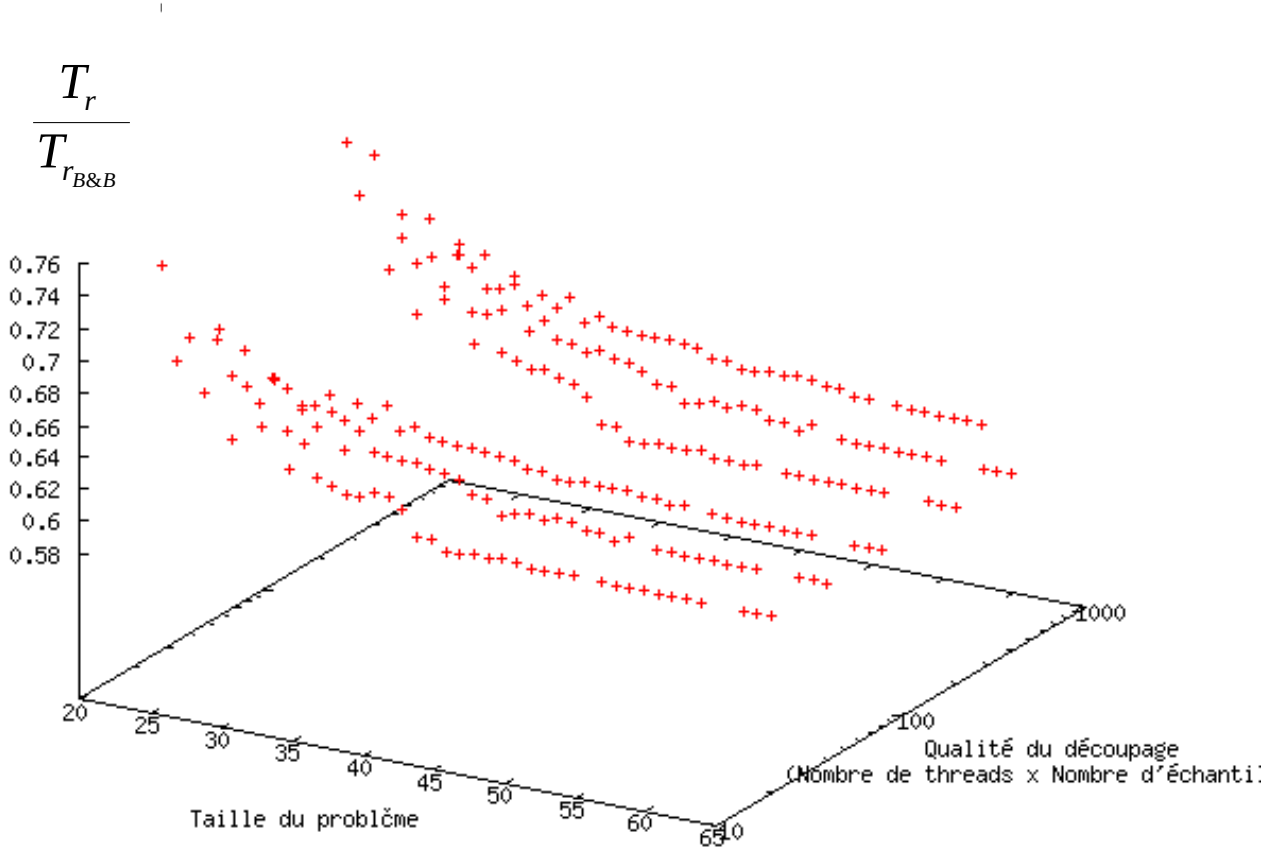
Résultats expérimentaux (1/6)

N-reines, 1 processus.



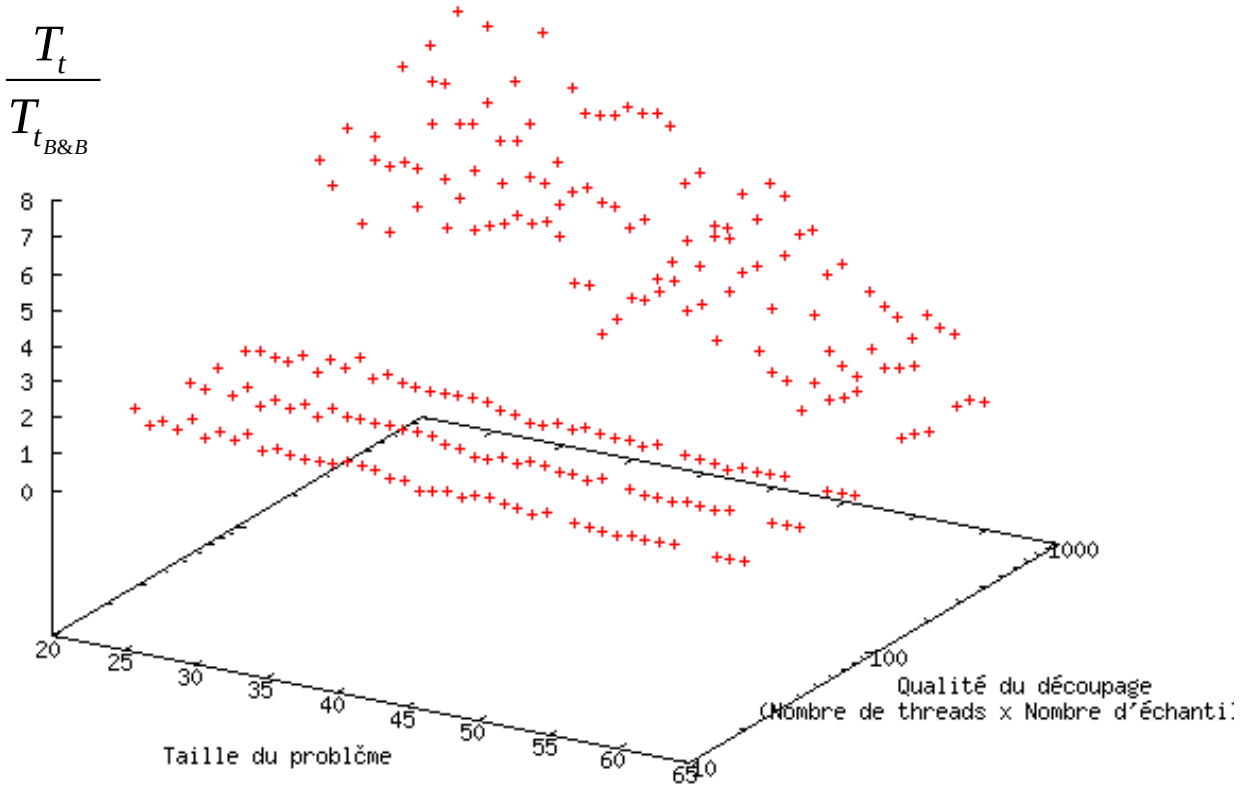
Résultats expérimentaux (2/6)

N-reines, 1 processus.



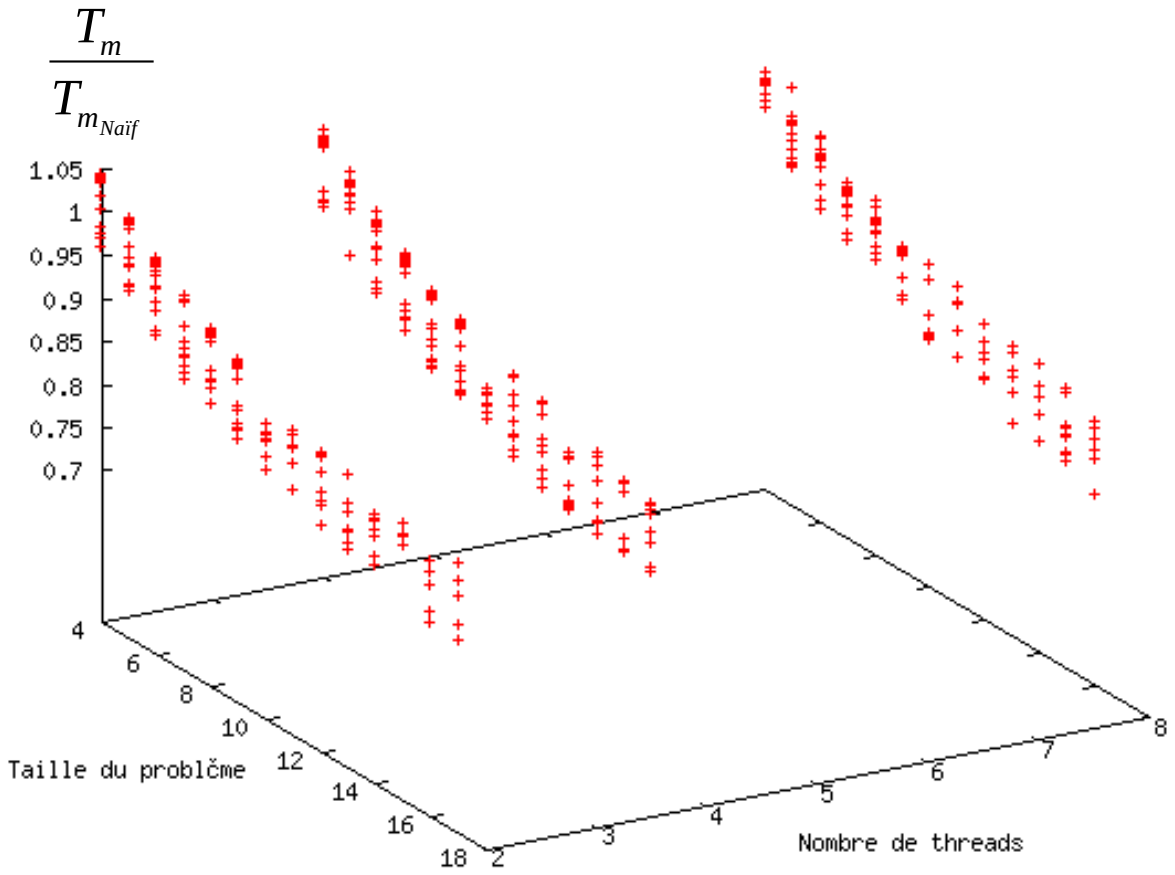
Résultats expérimentaux (3/6)

N-reines, 1 processus.

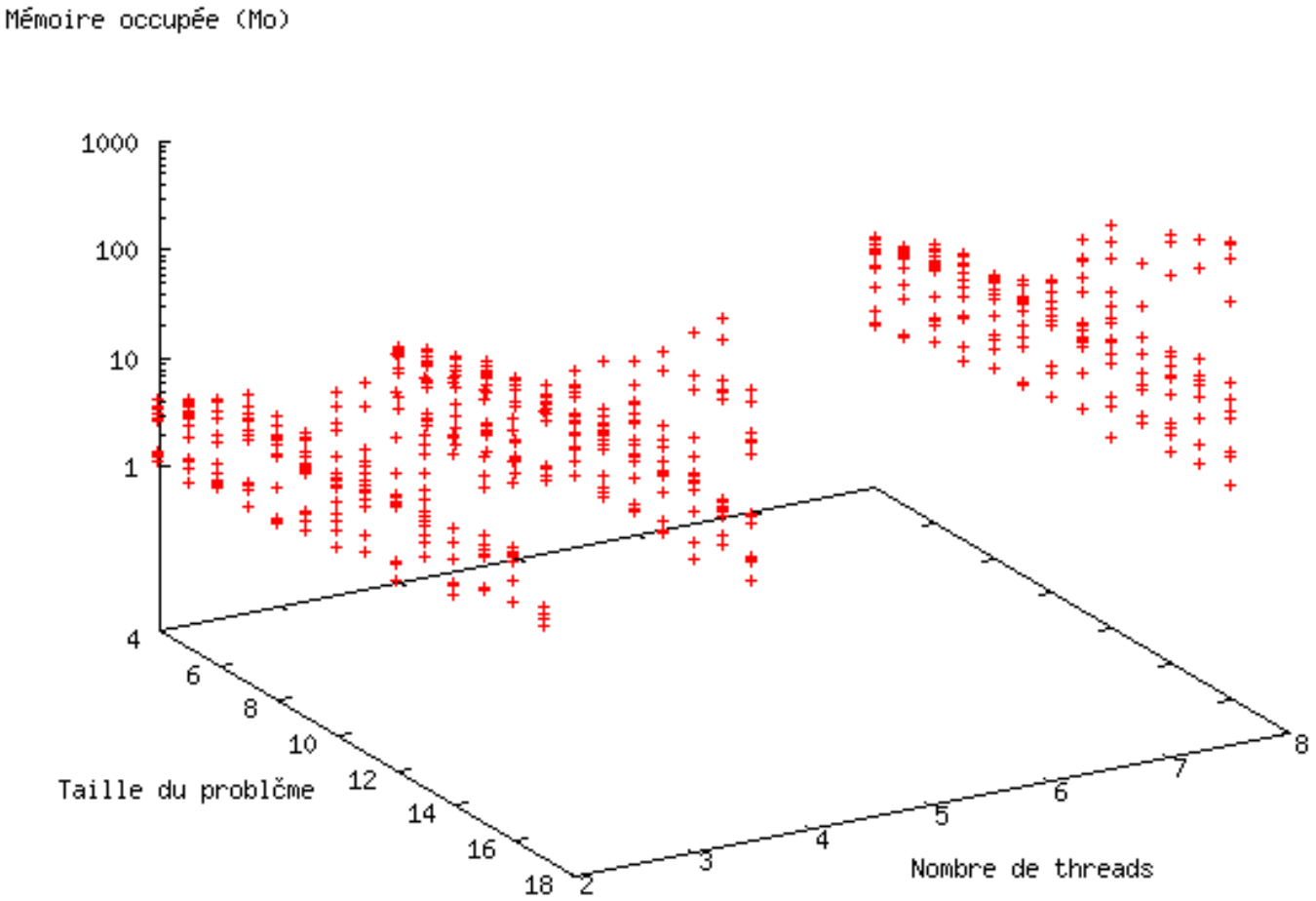


Résultats expérimentaux (4/6)

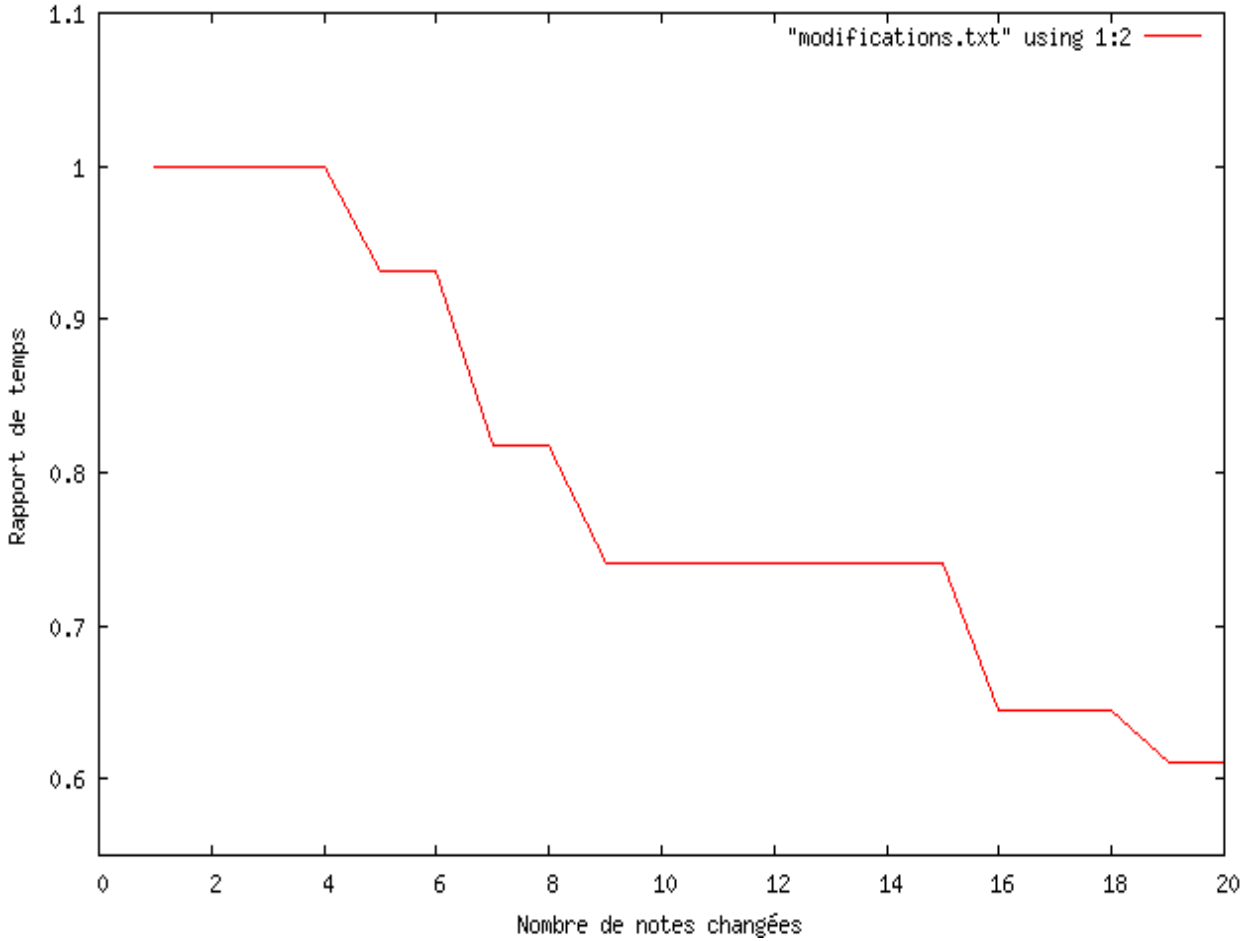
Fugue, multi-processus.



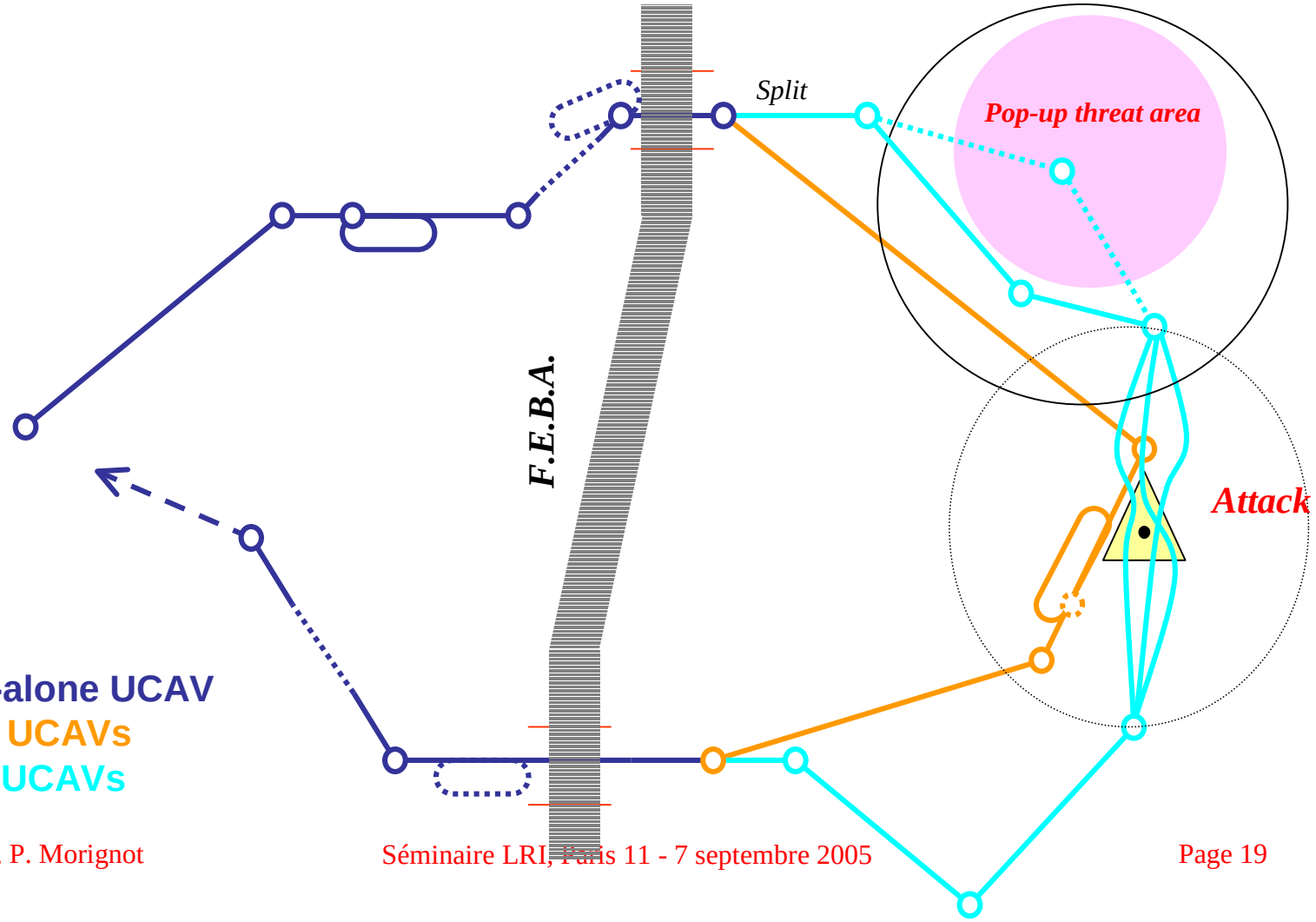
Résultats expérimentaux (5/6)



Résultats expérimentaux (6/6)



Application : drônes intelligents



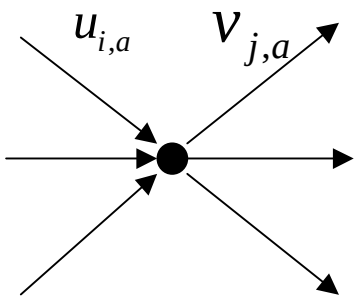
Application : modèle (sketch)

$$\forall n, \sum_i u_{i,n} = \sum_j v_{j,n} = p_k$$

$$\forall k, a_k \leq p_k$$

$$\forall k, \forall n, a_k = 1 \Rightarrow i_k = t_{k,n}$$

$$p_k = 1 \wedge p_{k'} = 1 \Rightarrow X_{k'} = X_k \oplus x_{k,k'}$$



max(probabilité de survie, probabilité de destruction)

Conclusion

- Estimateur stochastique + découpage + résolutions séparées avec communication.
- Complétude.
- Attention au temps de découpage → grands problèmes.
- Jusqu'à 30% plus rapide qu'une approche distribuée « naïve » (en 1/N).
- Travaux futurs :
  - Multi-processus réel → temps de communication.
  - Comparaison avec un découpeur aléatoire ?
  - Communication entre agents dès que le coût diminue.

Références

- A. Gorge, P. Morignot. *Une corrélation en programmation par contraintes distribuée*. Rapport Technique AXLOG, Arcueil, juin 2005, 10 pages. Soumis à RFIA'06.
- P. J. Modi, W. Shen, M. Tambe, M. Yokoo. *ADOPT: Asynchronous distributed constraint optimization with quality guarantees*. Artificial Intelligence, 161, pages 149-180, 2005.
- N. Prcovic, B. Neveu, P. Berlandier. *Distribution de l'arbre de recherche des problèmes de satisfaction de contraintes en domaines finis*. In Actes de la Deuxième Conférence Nationale sur la Résolution de Problèmes NP-Complets (CNPC'96), août 1996.