Parallel Discrepancy-based Search

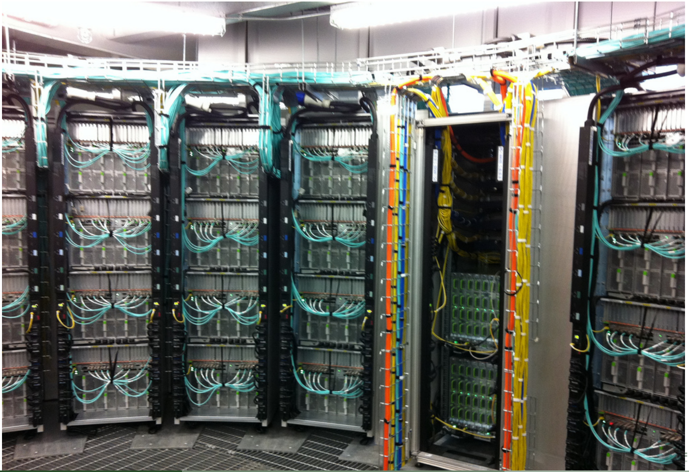
T. Moisan, J. Gaudreault, C.-G. Quimper

Université Laval, FORAC research consortium

February 21^{th} 2014

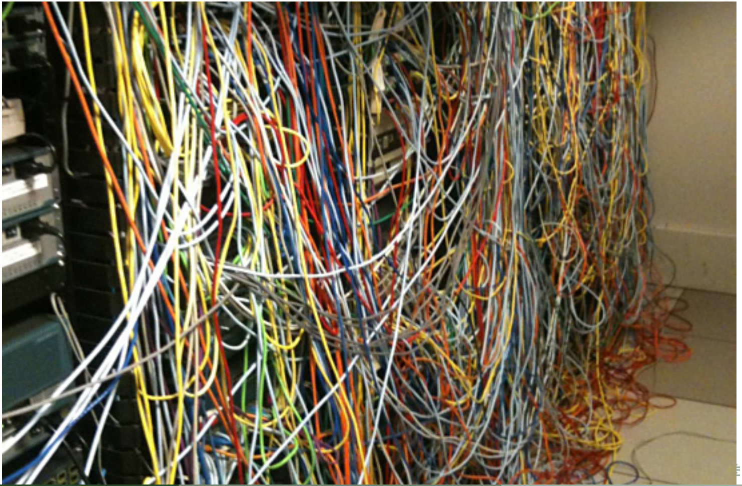
Upsides of Parallelization

- Parallel computing is a growing domain.
- Governments and industries are intensively investing in it.
- It is a huge opportunity for constraint programming.
- It can lead to unprecedented performances.



Downsides of Parallelization

- Parallelization is hard.
- Speedup can stall very fast due to communication.
- It is difficult to reproduce in a parallel environment the search strategies that work well sequentially.

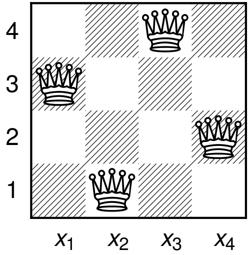


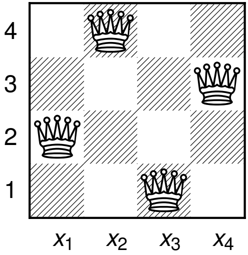
Outline

Constraint programming

- Uses variables with a finite domain;
- Uses specific constraints to modelize the problem:
  - All-different: variables must have different values;
  - Regular: variables must follow a defined regular expression.
- Each constraint includes a specialized algorithm to filter the search space;
- Can include an objective function.

Exemple



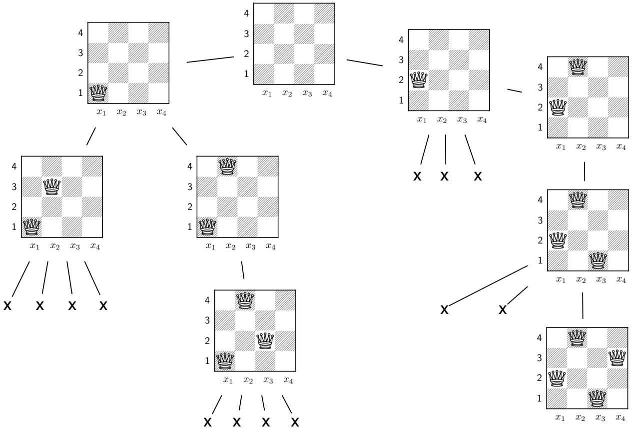


$$\mathrm{dom}(x_1) = \{1, 2, 3, 4\}$$
$$\mathrm{dom}(x_2) = \{1, 2, 3, 4\}$$
$$\mathrm{dom}(x_3) = \{1, 2, 3, 4\}$$
$$\mathrm{dom}(x_4) = \{1, 2, 3, 4\}$$

$$x_i \neq x_j \ \forall i < j$$
$$x_j - x_i \neq j - i \ \forall i < j$$
$$x_j - x_i \neq i - j \ \forall i < j$$

Espace de recherche

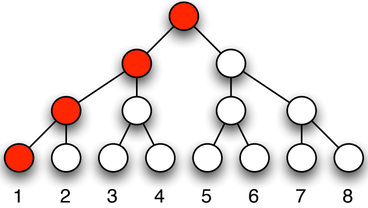
- Ensemble des assignations partielles des variables
  - Cet espace est typiquement représenté par un arbre n-aire.



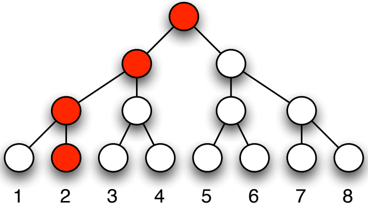
Outline

Depth-first Search (DFS)

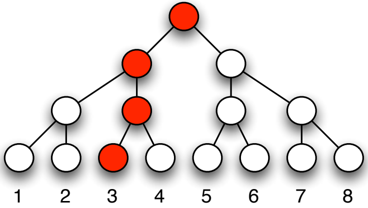
The value ordering heuristic function orders the children of a node from the most likely one to lead to a solution to the least likely one.



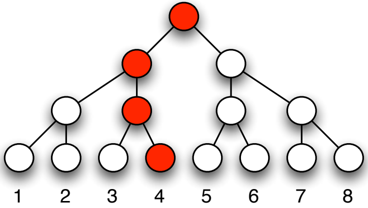
Depth-first Search (DFS)



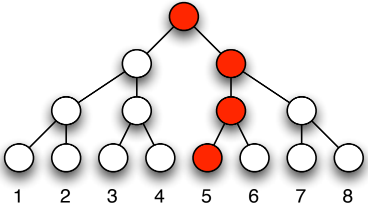
Depth-first Search (DFS)



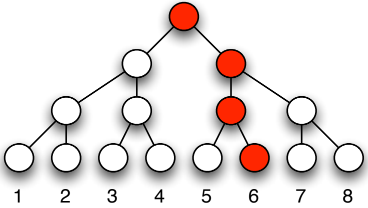
Depth-first Search (DFS)



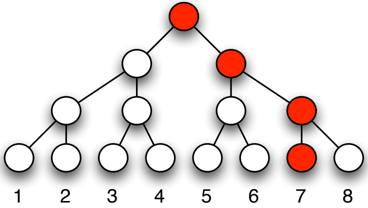
Depth-first Search (DFS)



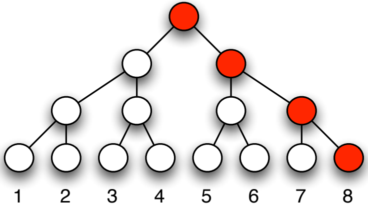
Depth-first Search (DFS)



Depth-first Search (DFS)



Depth-first Search (DFS)

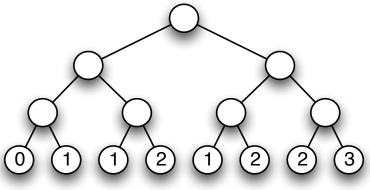


Outline

Limited Discrepancy Search (LDS)

Harvey and Ginsberg (1995)

The value ordering heuristic function orders the children of a node from the most likely one to lead to a solution to the least likely one. A discrepancy is a deviation from the first choice of the value ordering heuristic.

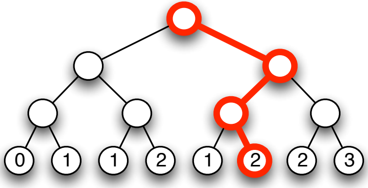


LDS proposes to visit the leaves in increasing order of discrepancy.

Limited Discrepancy Search (LDS)

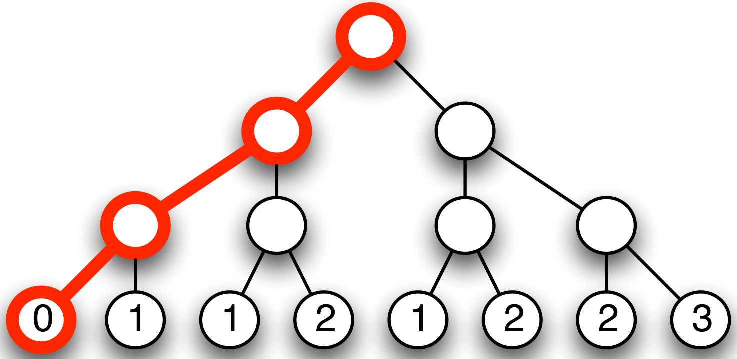
Harvey and Ginsberg (1995)

The value ordering heuristic function orders the children of a node from the most likely one to lead to a solution to the least likely one. A discrepancy is a deviation from the first choice of the value ordering heuristic.

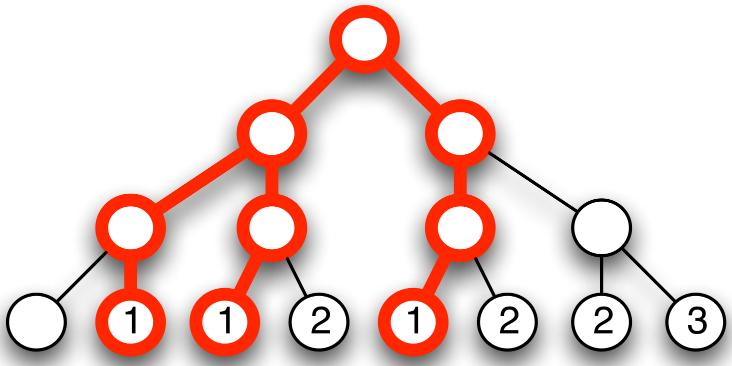


LDS proposes to visit the leaves in increasing order of discrepancy.

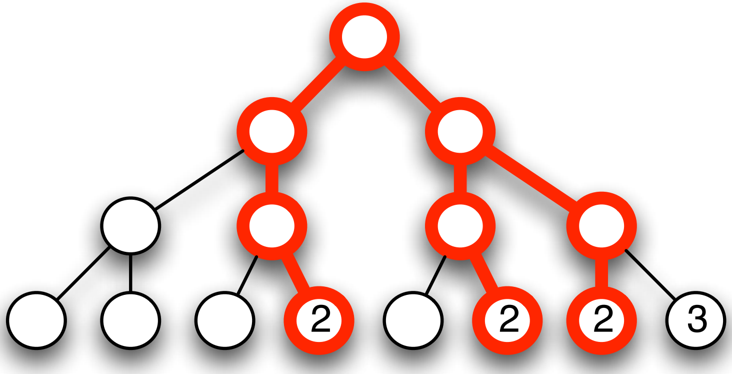
LDS Iterations 0 (0 discrepancy)



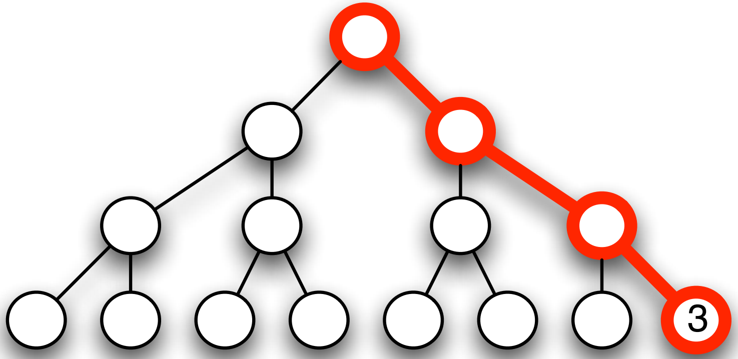
LDS Iterations 1 (1 discrepancy)



LDS Iterations 2 (2 discrepancies)



LDS Iterations 3 (3 discrepancies)



LDS Overhead versus DFS

- Nodes in a binary tree with n variables: $2 \cdot 2^n - 1$
- Node visits with a DFS: $2 \cdot 2^n - 1$
- Node visits with a LDS: $4 \cdot 2^n - n - 3$
- By the time DFS visits the entire tree, LDS will visit half of the leaves.
- The leaves that LDS visits are those that have fewer than n/2 discrepancies.
- If the value ordering heuristic is as good as a random heuristic, LDS finds a solution by the time DFS completes.

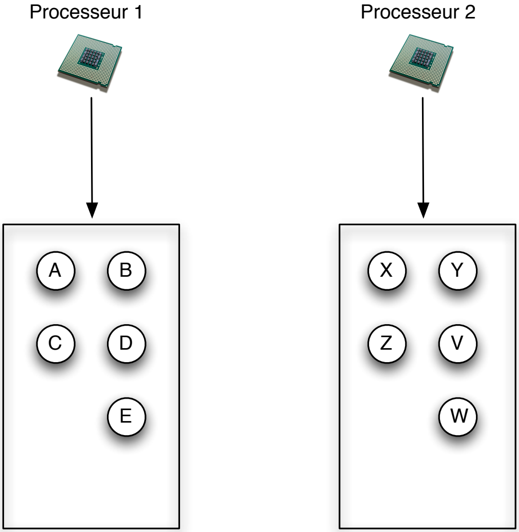
Outline

Portfolio

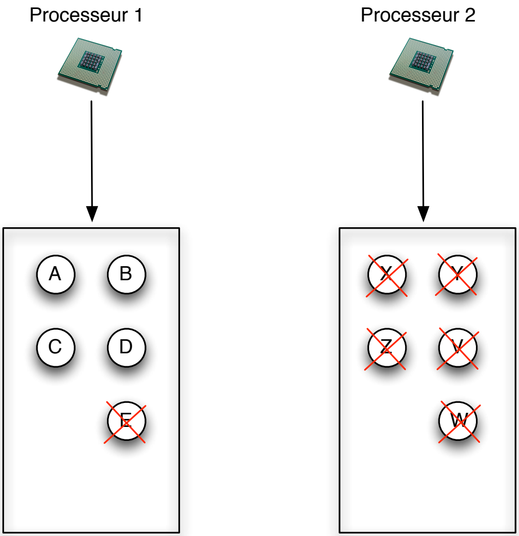
- Il s'agît d'une course entre les processeurs pour obtenir une solution.
- Chaque processeur recherche une solution avec un algorithme qui lui est propre
- Les processeurs peuvent s'aider en échangeant des informations.
- Il est difficile d'étendre à un nombre arbitraire de processeurs.
  - Il faut définir des paramètres de recherche pour chaque processeur.
  - Ces paramètres sont souvent établis manuellement.
  - La paramétrisation est fortement lié au problème à résoudre.

Work-stealing

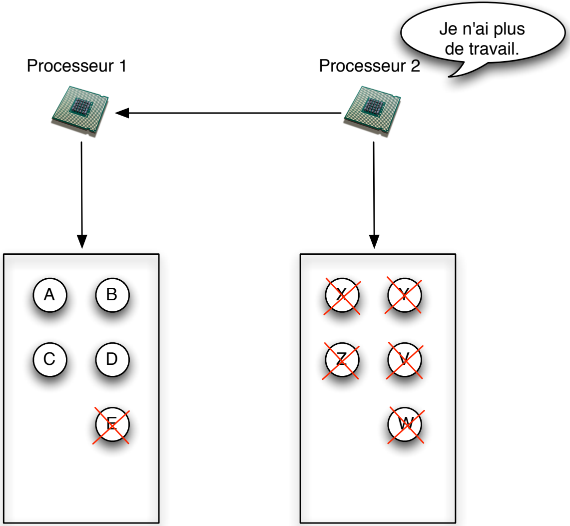
Xie et Davenport



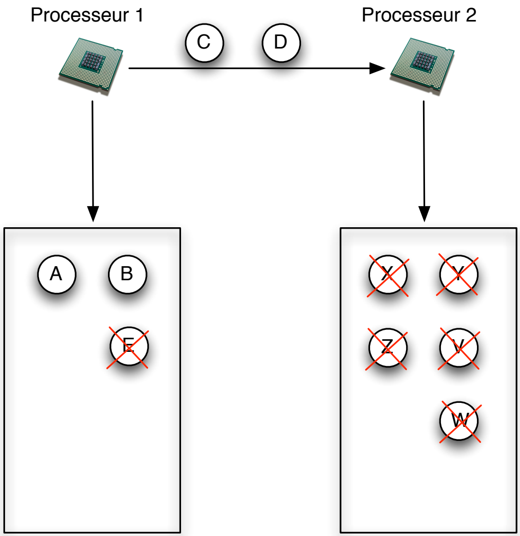
Work-stealing



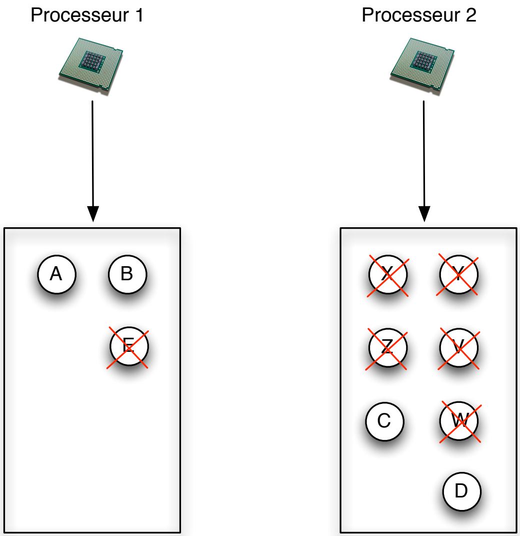
Work-stealing



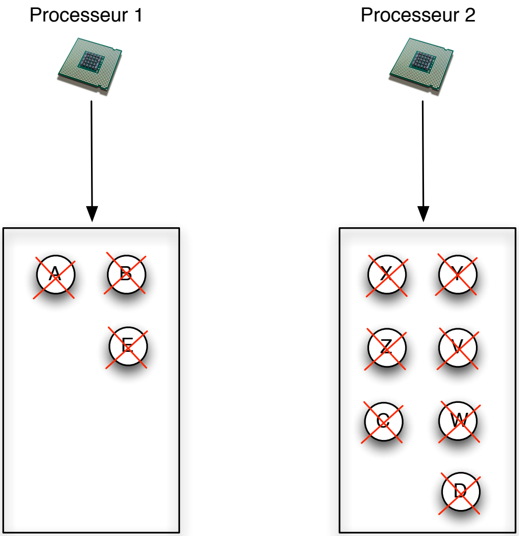
Work-stealing



Work-stealing



Work-stealing



Division de l'espace de recherche

- Approche très courante
- Divise théoriquement la charge de travail également
- Peu probable que les sous-arbres soient de même taille
  - Ne peut pas être utilisé sur un grand nombre de processeurs
  - Problème NP-Difficile
- Si les sous-arbres ne sont pas de même taille
  - Les processeurs n'auront pas la même charge de travail
  - La durée totale de la tâche sera plus longue

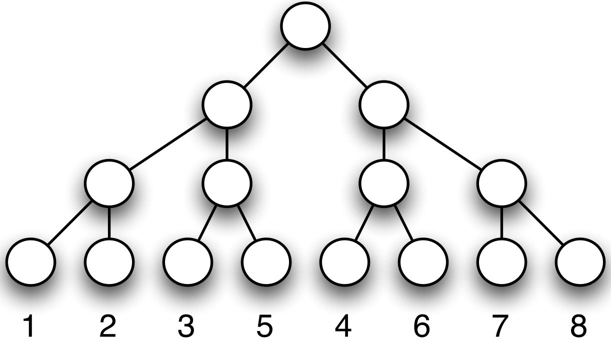
Outline

Our Objectives

- Having a search strategy that performs well sequentially, we want to parallelize it with four main goals in mind:
  - Search strategy preservation
  - Workload balancing
  - Robustness
  - No communication

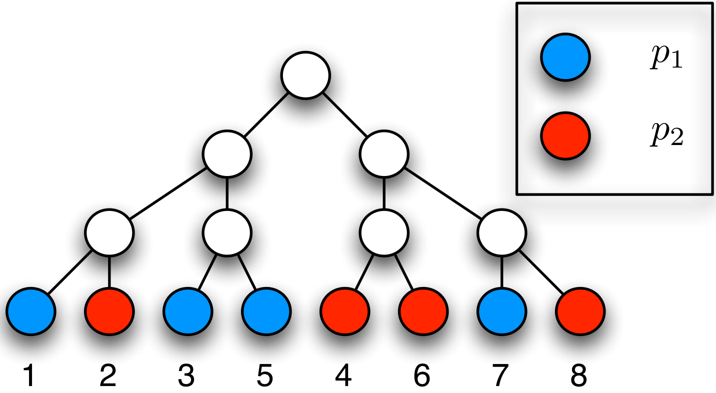
Intuition behind PDS

- Each leaf is assigned to a processor.
- The leaves are implicitly assigned in a round-robin fashion.
- To split the task, let's say that p_1 takes odd leaves and p_2 the even leaves.

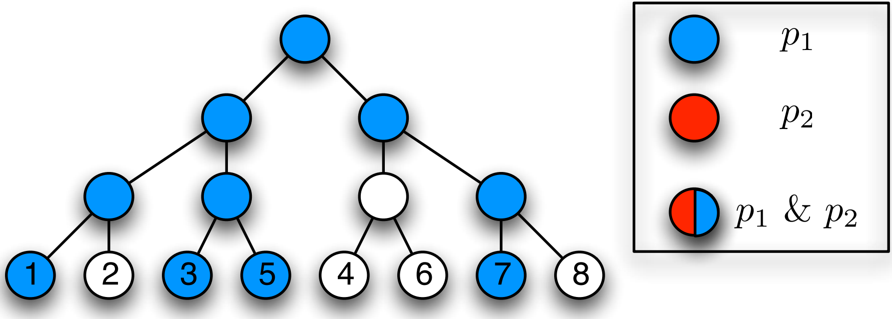


Intuition behind PDS

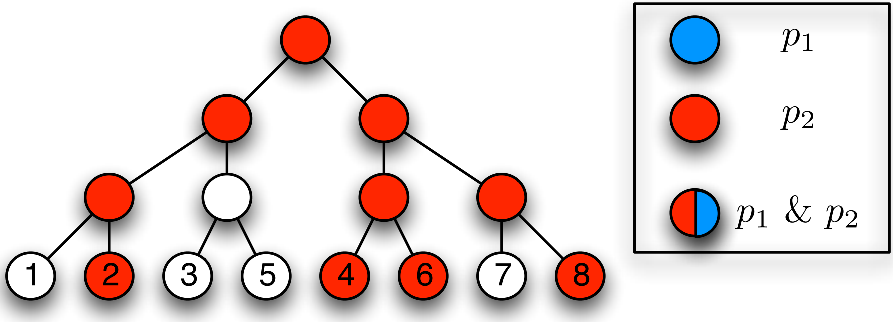
- Each leaf is assigned to a processor.
- The leaves are implicitly assigned in a round-robin fashion.
- To split the task, let's say that p_1 takes odd leaves and p_2 the even leaves.



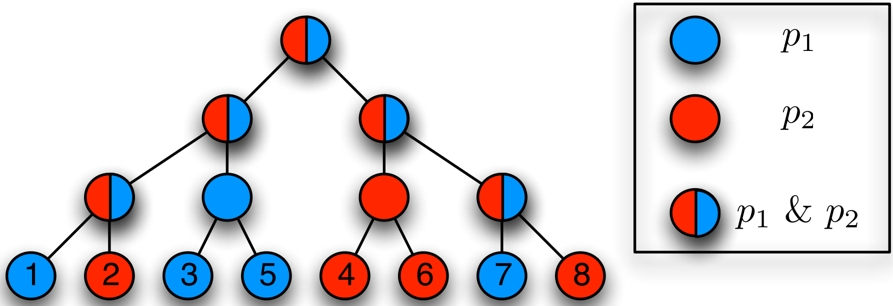
PDS overview



PDS overview

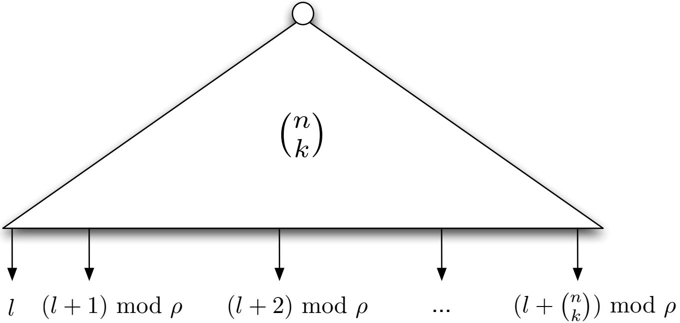


PDS overview

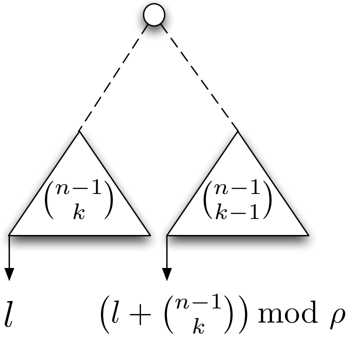


Implementing PDS

- p : its processor id
- ρ : the number of processors
- n : the number of variables in the subtree
- k: the number of required discrepancies
- l : the processor id implicitly assigned to the leftmost leaf
- The processor will branch in the subtree if $(p - l) \bmod \rho < \binom{n}{k}$



Branching in PDS

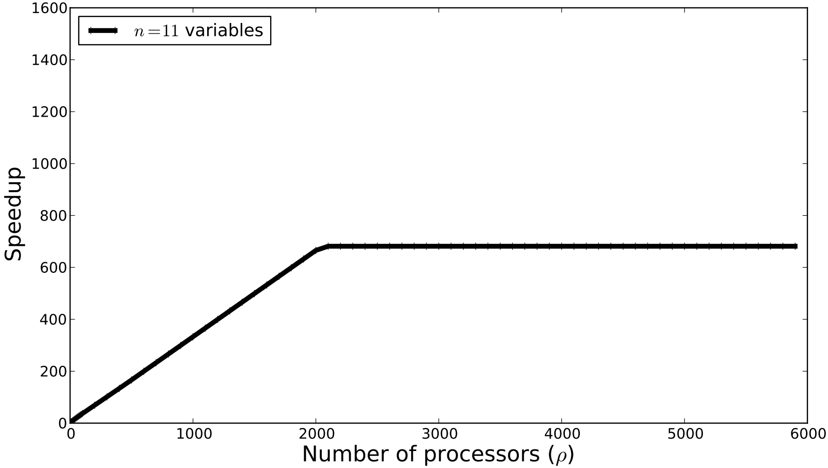


Outline

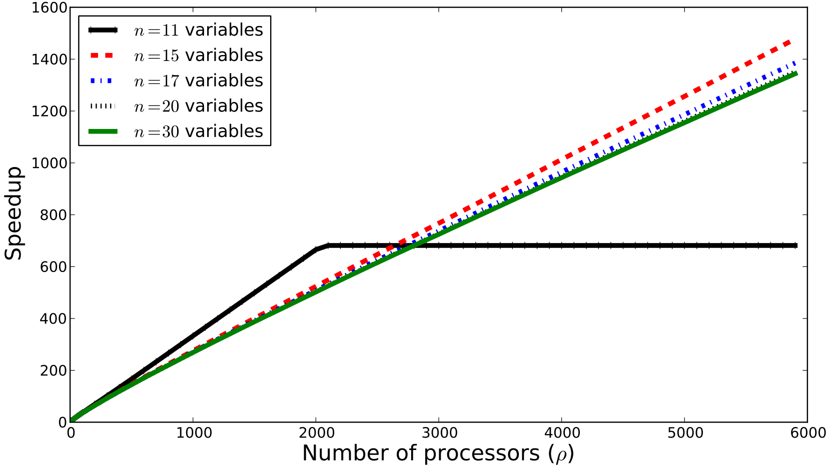
PDS overhead versus LDS

Processors	Node visits	Overhead	Speedup
1	$4 \cdot 2^n - n - 3$	-	-
2	$5 \cdot 2^n - 2n - 4$	25%	1.61
3	$5.75 \cdot 2^n - 3n - 5$	43%	2.08
...	...	...	...
ρ	$2^n + 2^n \sum_{i=1}^{n} \sum_{k=0}^{i} \frac{1}{2^i} \min(\rho, \binom{i}{k})$		

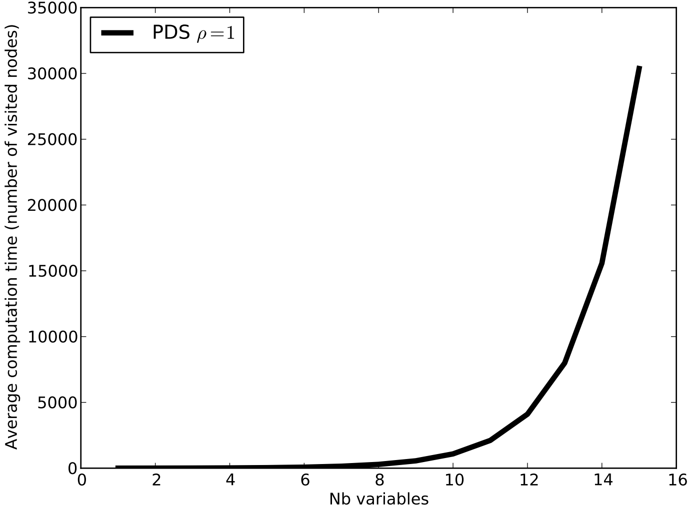
Speedup for some number of processors



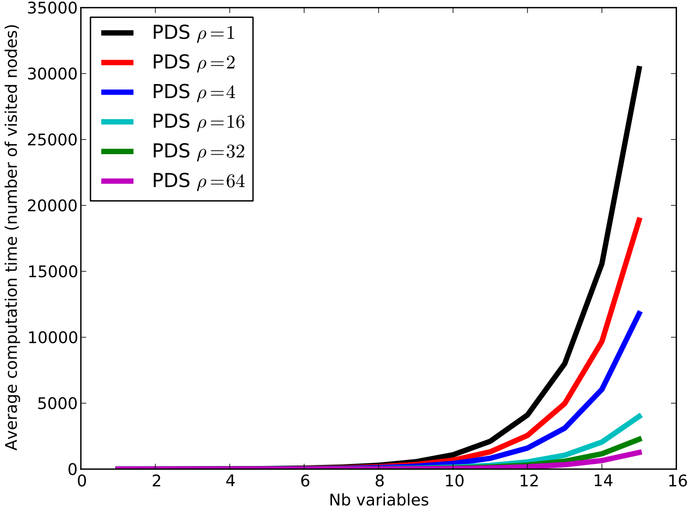
Speedup for some number of processors



Average computation time to find a solution according to the number of variables



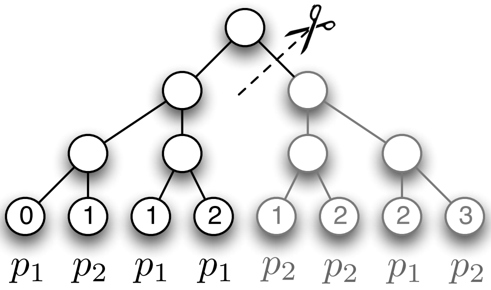
Average computation time to find a solution according to the number of variables



Balancing

Theorem

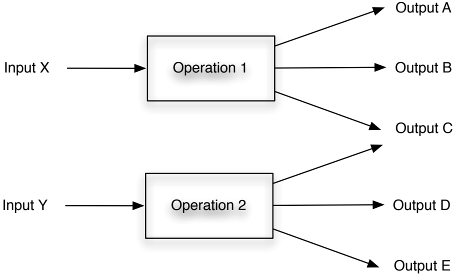
Let n be the number of variables in the problem. If a branch is cut from the search tree, the number of leaves removed from the workload of each processor differs by at most n.



Outline

Planning and scheduling wood finishing operations

- We minimize order lateness.
- The production is done with one-to-many operations.
- There is multiple ways to produce the same product.



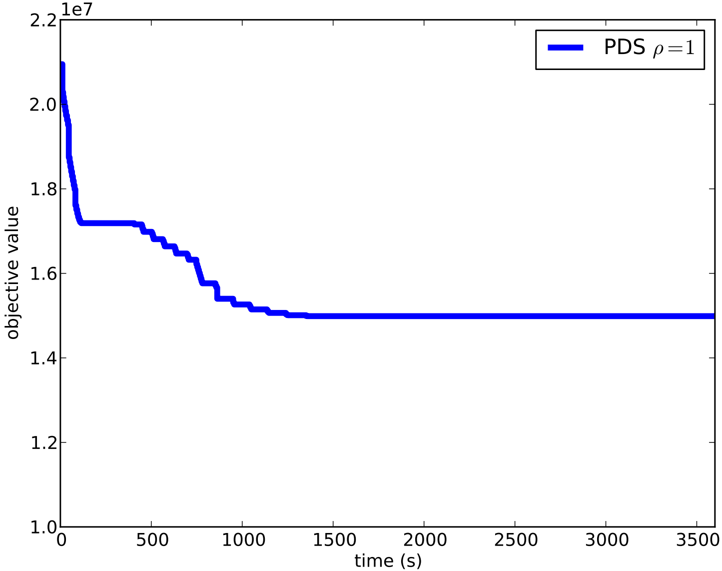
- There are setup constraints that restrict sequencing of the operations.
- This is a problem for which LDS was efficient thanks to a specialized value ordering heuristic (Gaudreault et.al (2010)).

Instances from a Canadian Forest-Product Company

We worked with industrial instances:

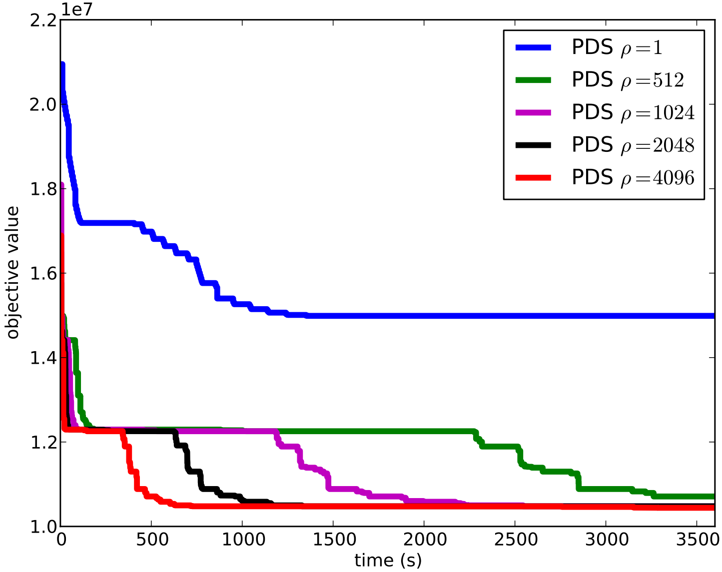
- 65,142 variables
- 50,238 constraints
- 42 discrete decision variables whose domains have cardinality 6
  - Used LDS/PDS to fix those variables.
  - Once the discrete decision variables are known, the remaining continuous variables define a linear program that can be easily solved to optimality.
- 4200 continuous decision variables
  - Linear program solved with Cplex.

Results



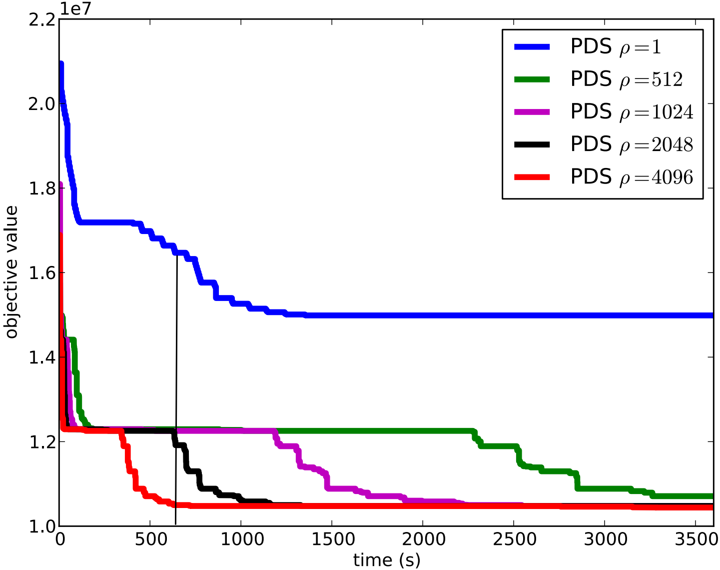
1e7
2.2
2.0
1.8
1.6
1.4
1.2
1.0
objective value
0
500
1000
1500
2000
2500
3000
3500
time (s)
PDS $\rho = 1$

Results



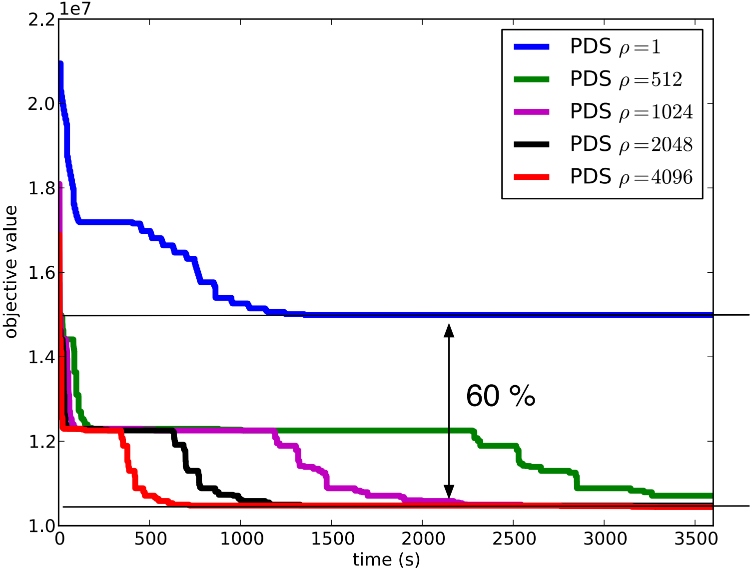
1e7
2.2
2.0
1.8
1.6
1.4
1.2
1.0
objective value
0
500
1000
1500
2000
2500
3000
3500
time (s)
PDS $\rho = 1$
PDS $\rho = 512$
PDS $\rho = 1024$
PDS $\rho = 2048$
PDS $\rho = 4096$

Results



1e7
2.2
2.0
1.8
1.6
1.4
1.2
1.0
objective value
0
500
1000
1500
2000
2500
3000
3500
time (s)
PDS $\rho = 1$
PDS $\rho = 512$
PDS $\rho = 1024$
PDS $\rho = 2048$
PDS $\rho = 4096$

Results



1e7
2.2
2.0
1.8
1.6
1.4
1.2
1.0
objective value
0
500
1000
1500
2000
2500
3000
3500
time (s)
PDS $\rho = 1$
PDS $\rho = 512$
PDS $\rho = 1024$
PDS $\rho = 2048$
PDS $\rho = 4096$
60 %

Conclusion

- Theoretical analysis of DFS versus LDS
- Parallel Discrepancy-based Search (PDS)
  - LDS search strategy preservation
  - No communication
  - Intrinsic load balancing
  - Robust to hardware failures
- Theoretical analysis of this parallelization.
- Experimental results on large industrial instances with up to 4096 processors.