Improving branch-and-bound exploration for Open-Shop problems

Narendra Jussien
Christelle Guéret
Ecole des Mines de Nantes
France
Outline

• Open Shop problems
• A branch-and-bound exploration for OS
• What’s wrong with chronological backtracking
• Improving tree searches
• First results
• Future works
The Open Shop problem

- Data: \( n \) jobs composed of \( m \) tasks to process through \( m \) machines

- Constraints
  - a machine can process only one task at a time
  - two tasks of a same job cannot overlap

- Criterion: smallest makespan

- Complexity: NP-hard for \( m \geq 3 \)
Solving Open Shop problems

- Good heuristics have been developed
  - Gueret & Prins 98

- A few branch and bound exploration
  - the best one: Brucker et al. ??
  - Some 7x7 problems are still unsolved
Brucker et al. B&B

- Depth-first search algorithm
- Node creation
  - build a heuristic solution
  - compute a critical path on that solution
  - fix disjunctions on that critical path
    - branching scheme of Grabowski et al.
- Node propagation
  - disjunctions are propagated using *immediate selections* from Carlier & Pinson
Drawbacks of DFS

Machine 1

Machine 2

Machine 3

Machine 4

UB = 25

machine 1
J1 → J2 → J3 → J4

machine 2
J1 → J2 → J3 → J4

machine 3
J1 → J2 → J3 → J4

machine 4
J1 → J2 → J3 → J4

16/08/1999
N. Jussien and C. Guéret
Drawbacks of DFS

• Unnecessary explorations

• Thrashing
  – doing over and over the same unusefull explorations
Overcoming the drawbacks

• Basic principle
  – recording \textit{information} during the search
  – using that information to compute a \textit{relevant} backtrack point when encountering a contradiction
  – using that information to determine \textit{reusable} parts of previous computations

• Basic tool
  – \textit{explanation} : set of nodes that leads to the attached action
    • modification of head or tail of a task
    • inferred precedence constraint
Using explanations

• Backtrack occurs when rejecting a node

• What can be done
  – there is no valid start date for a given task
    • Explanation E: union of the explanations of all modifications made on the associated variable
  – all the children of a given node have been tested
    • Explanation E: union of the explanations of each unsuccessful child

• The most recent relevant backtrack point is the most recent node in E

Intelligent Backtracking
A more complete usage

- Saving computation
  - Let \( n \) be the chosen node for backtracking
  - every explanation that does not contain \( n \) is still valid
  - backtracking is replaced by path modification

- Remaining complete
  - some more information is stored

**Dynamic Backtracking**
Graphical comparison

16/08/1999  N. Jussien and C. Guéret
Complexity issues

- spatial complexity overhead: $O(e.n.d)$
  - $n = \text{number of variables} + \text{number of precedences}$
  - $d = \text{depth of the search tree}$
  - $e = \text{size of an explanation} (O(d))$

- time complexity overhead: polynomial
  - union of a polynomial number of explanations of polynomial size during constraint propagation
Computational results

- Tests performed on Taillard’s problems
Computational results

- **General observations**
  - Intelligent Backtracking always improves the search up to 90% for some problems
  - Time overhead is widely compensated by the improvements in number of developed nodes
  - Dynamic Backtracking is more stable for series of similar problems: it depends less on the data

- **Specific results**
  - IB allowed to solve a 10x10 open problem
  - DB can greatly improve results but can also get caught in the *bad* part of the search space
Conclusion and Future works

- **Recording information** during the search may greatly improve obtained results

- Dynamic Backtracking is promising but too restricted by completeness constraints
  - currently testing an heuristic method based on that idea
    - taking rid of the completeness constraints
    - using heuristic techniques to determine the next modification
    - sort of tabu search on the paths