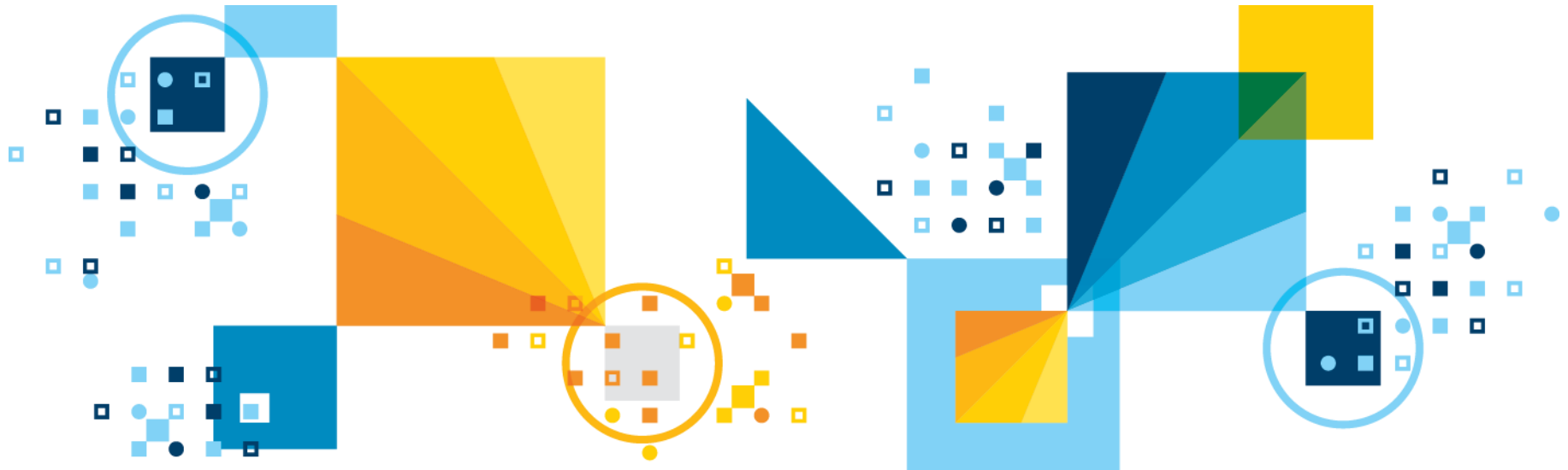


Andrea Tramontani
CPLEX Optimization, IBM
CWI, Amsterdam, June 12, 2018

Heuristics in Commercial MIP Solvers

Part I (Heuristics in IBM CPLEX)



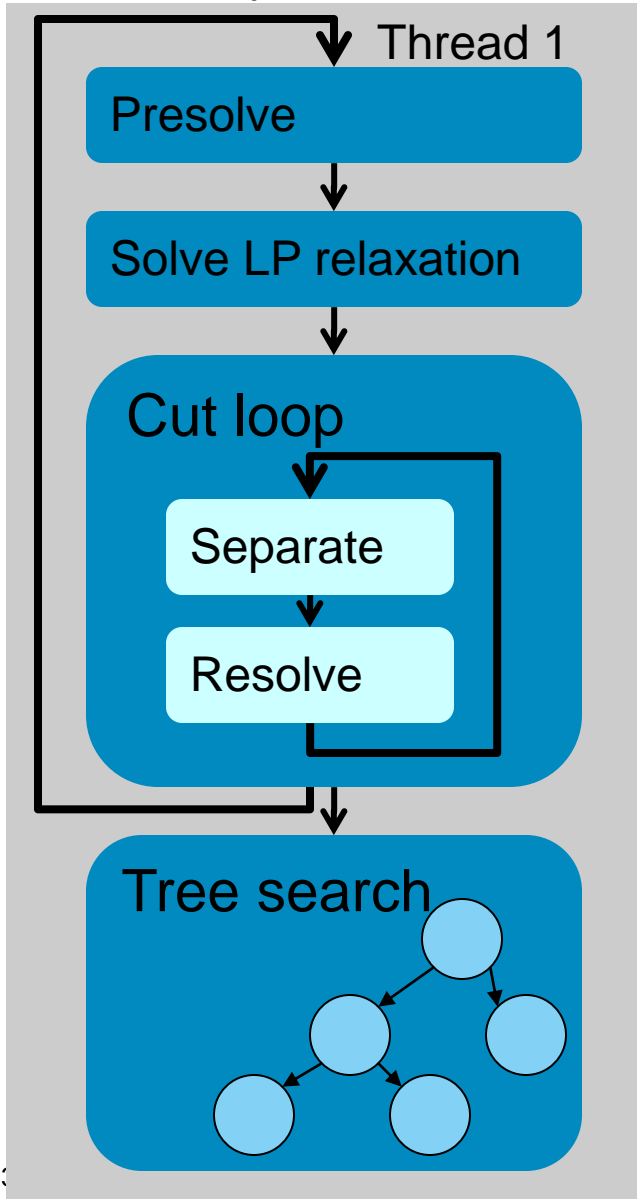
Agenda

- CPLEX Branch-and-Bound (B&B)

- Primal heuristics in CPLEX
 - Overview and classification
 - Some examples
 - Diving heuristics
 - SubMIP heuristics
 - Tabu search heuristic for 0-1 IPs
 - Heuristic manager

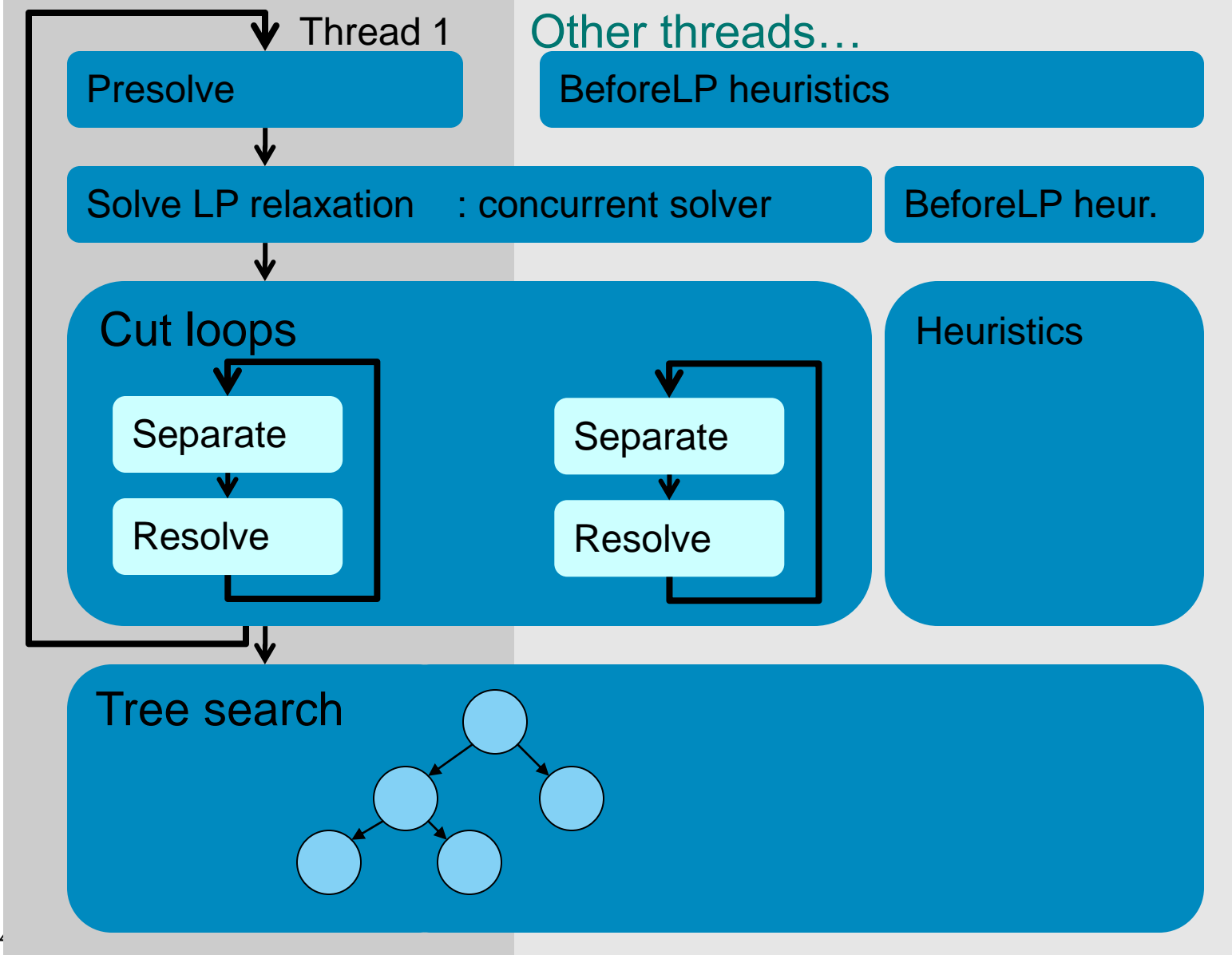
- Performance analysis
 - Performance impact of heuristics in solving problems to optimality

CPLEX sequential MIP solver



$$\begin{aligned} (MIP) \quad & \text{Minimize} \quad z = c^T x \\ & \text{Subject to} \quad Ax = b \\ & \quad \quad \quad l \leq x \leq u \\ & \quad \quad \quad \text{some or all } x_j \text{ integer} \end{aligned}$$

CPLEX parallel MIP Solver



Primal heuristics in one slide

- What are they?
 - **Incomplete methods** that simply look for feasible solutions
 - No guarantees of any kind
 - Not always predictable (short) running time
 - Must run within a **controlled environment**

- Why do we need them?
 - Prove feasibility of the model
 - **Speed up search**
 - Primal bound needed for pruning and reduced cost fixing
 - Sometimes good enough for **practical purposes**
 - Often optimization is stopped when gap is small enough (not yet 0)
 - Fundamental when problems can't be solved to optimality (e.g., optimality proof is unpractical or strict work limits are imposed)

Starting heuristics vs. improving heuristics

- Starting heuristics
 - Do not need any feasible solution available
 - But can implicitly exploit an UB if available (e.g., diving heuristics)
 - Relevant examples
 - Before LP heuristics
 - Rounding heuristics
 - Tabu search for 0-1 IPs (inspired to WalkSat paradigm)
 - **Diving heuristics**
 - Feasibility pump (Fischetti et al., 2003)

- Improving heuristics
 - Explore neighborhoods of feasible solutions to improve on the incumbent solution
 - Relevant examples
 - Tabu search for 0-1 IPs (inspired to WalkSat paradigm)
 - **RINS (Danna et al., 2005)**
 - Genetic Algorithm (Rothberg, 2007)
 - Local branching (Fischetti and Lodi, 2003)

Before LP heuristics vs. after LP heuristics

- Before LP heuristics
 - Run sequentially before solving the first LP relaxation or concurrently to the first LP solve
 - **Cannot take advantage from the knowledge of an LP solution**
 - Almost all of them are starting heuristics
 - Sequential heuristics
 - Must be cheap (**very strict work limit**)
 - Concurrent heuristics
 - Do not need strict work limit (**killed anyway when the first LP solve is done**)
 - Relevant examples
 - Fix and propagate (no LP solves)
 - Simple local search (no LP solves)
 - Tabu search for 0-1 IPs (inspired to WalkSat paradigm)
 - Fix and solve subMIP
 - Zero-objective subMIP

Before LP heuristics vs. after LP heuristics (Cont. d)

- After LP heuristics
 - Can take advantage from the **knowledge of an LP solution**
 - Allowed to be **more expensive**
 - Can solve one or more LP of same size of the full LP relaxation
 - Relevant examples
 - Rounding heuristics
 - Tabu search for 0-1 IPs
 - Diving heuristics
 - RINS (Danna et al., 2005)
 - Genetic Algorithm
 - Zero-objective subMIP

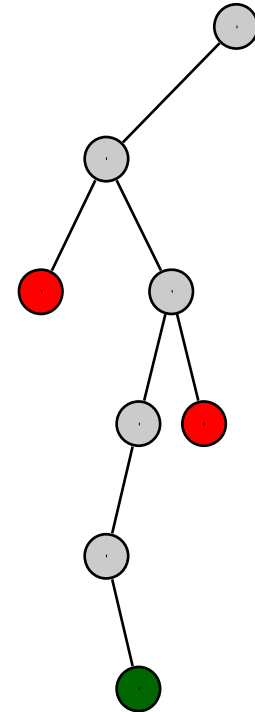
General purpose vs. special purpose heuristics

- General purpose heuristics
 - Do not need the problem to have any specific structure

- Special purpose heuristics
 - **Tabu search for 0-1 IPs**
 - Specialized heuristics for **set covering/partitioning** problems
 - Specialized B&B for problems with a big **set packing** components
 - No LP solve
 - Branching, node selection, constraint propagation entirely based on clique table
 - Inspired to Rapid Learning techniques (Berthold et al., 2010)

Heuristic catalog – Diving heuristics

- Given the fractional solution x of a certain B&B node
 - **Simulate Depth First Search** (DFS) with a special “branching” strategy:
 - Change variable bound(s)
 - Propagate constraints
 - Resolve LP
 - **Different variants** (from cheap to expensive) implemented:
 - Alternative strategies/scores to select the variable bound(s) to change
 - Different frequencies to resolve the LPs after propagation
 - Different level of backtracking allowed
- Very cheap version (no LP solve) applied **also as before LP heuristic**
- Heavily applied at the **root node** and in the **search tree**

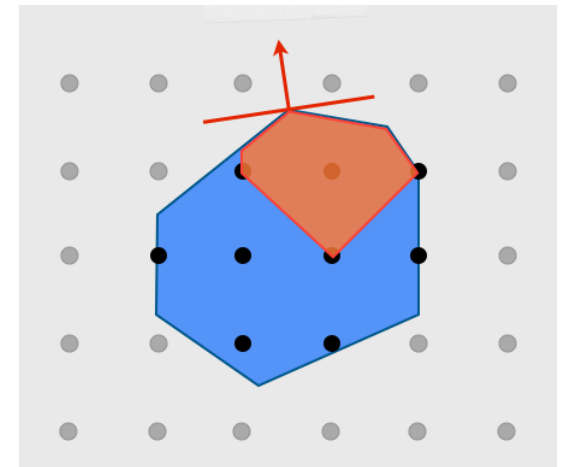


Heuristic catalog – SubMIP heuristics

- RINS (Danna et al., 2005)
 - Explore a neighborhood given by the incumbent solution and the current fractional solution
 - **Different variants** implemented (from cheap to expensive)
 - Applied at the **root node** and in the **search tree**

- Genetic Algorithm (GA) (Rothberg, 2007)
 - Generate (several) partial solution vectors by applying crossover and mutation operators to (some of) the feasible solutions available in the pool
 - For each partial solution vector v
 - Fix variables to the values of v
 - Solve the corresponding subMIP
 - Very expensive
 - **Sporadically applied** during tree search in default setting
 - Bulk of **solution polishing** feature

- Zero-objective subMIP
 - Remove objective function and solve subMIP
 - Can be expensive, but also applied as beforeLP heuristic (with proper work limits)



Heuristic catalog – Tabu Search (TS) heuristic for 0-1 IPs

```
Guess an infeasible 0-1 vector  $x$ ;  
Compute score  $s(x)$  that measures infeasibility of  $x$ ;  
while ( $x$  not feasible && work limit not reached) {  
    Select variable  $x_j$  to flip in  $x$   
        to obtain  $x'$  with smallest possible  $s(x')$ ;  
    Flip  $x_j$  and mark the flip as tabu;  
    Update  $x$  and  $s(x)$ ;  
}
```

- Different types of **randomization** applied:
 - Tabu tenure of each flip/move
 - Subset of flips/moves that are evaluated at every iterations
- Different ways to select the initial x vector:
 - A random vector
 - The incumbent solution vector (when used as improving heuristic)
 - ~~Rounding of solution of LP relaxation~~ (sounds natural, but **not done**)
- Called as **beforeLP heuristic** and at the **root node**

Heuristic manager

- CPLEX implements around 50 primal heuristics
 - Different variants of the same heuristic are considered as different ones

- The heuristic manager
 - Keeps **statistics** for every heuristic
 - #calls, success rate, deterministic time spent, ...
 - Decides
 - **Frequency of call** for every heuristic
 - **Work limit** for every heuristic call
 - Main goals:
 - Ensure some diversification
 - Favor heuristics that appear to be more effective
 - Make sure that, in the long run, the overall time spent in all heuristics is at most a certain fraction of the whole running time

Performance Analysis

Main building blocks: Measuring performance impact

- How important is each component?

Compare runs with feature turned on and off

- Solution time degradation (geometric mean)
- # of solved models
 - Essential or just speedup?
- Number of affected models
 - General or problem specific?

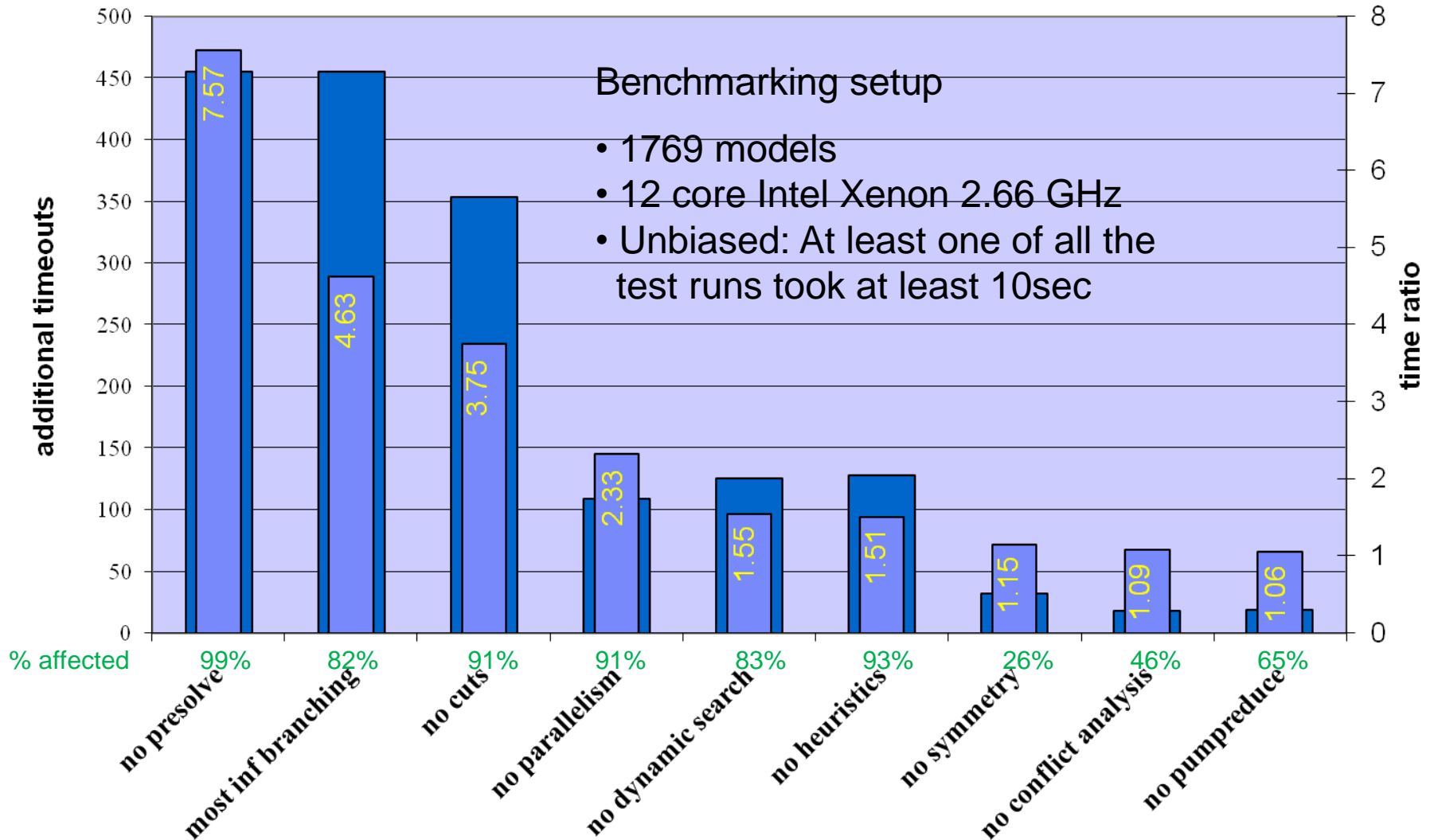
- Experiments conducted with CPLEX 12.5.0 (2012)

- Several features not available yet, e.g.,
 - L&P cuts, Parallel cut loop, TS heuristic for 0-1 IPs, other special purpose heuristics, ...

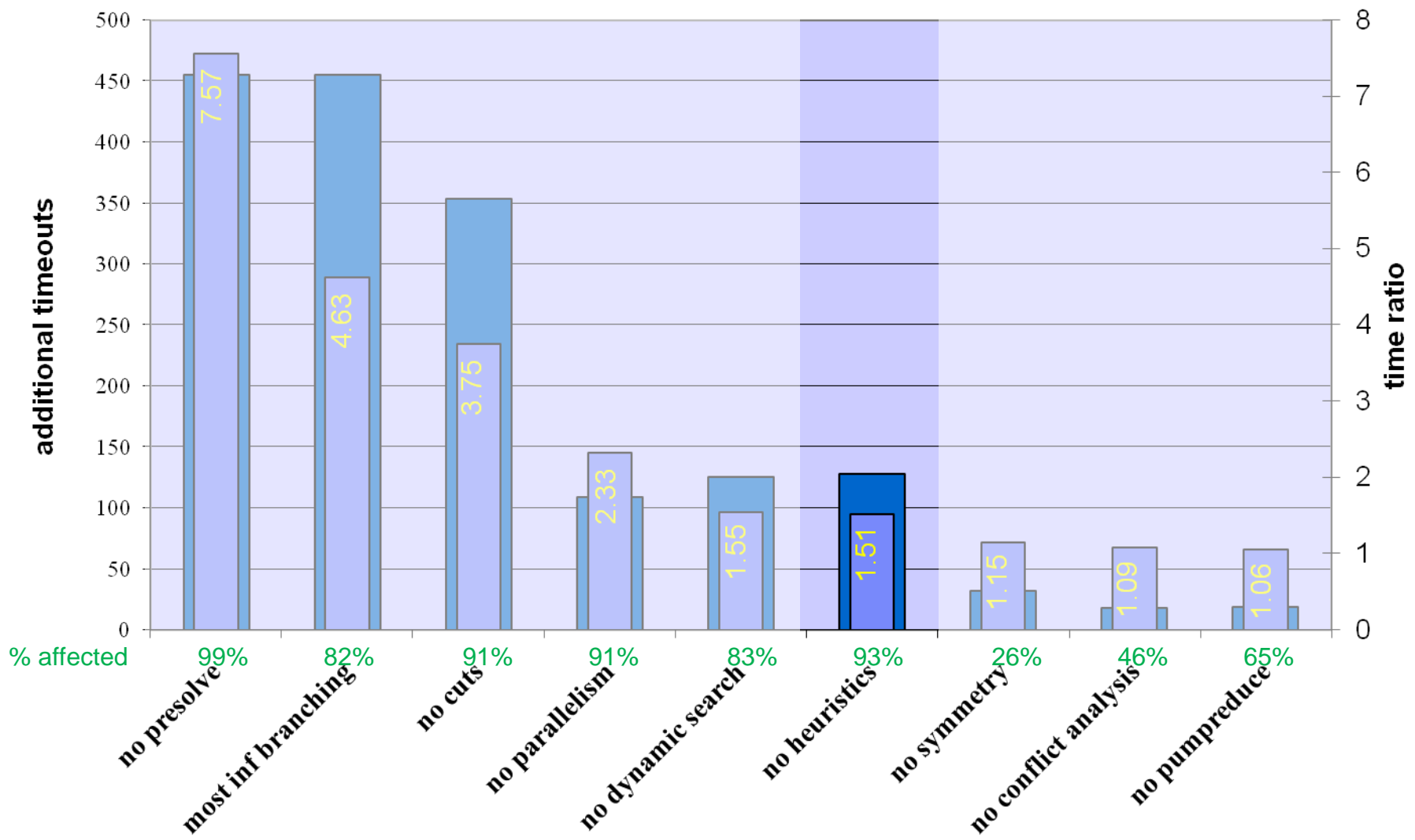
- More detailed analysis in:

T. Achterberg and R. Wunderling, “Mixed Integer Programming: Analyzing 12 Years of Progress”, in: Jünger and Reinelt (eds.) Facets of Combinatorial Optimization, Festschrift for Martin Grötschel, pp.449-481, Springer, Berlin-Heidelberg (2013)

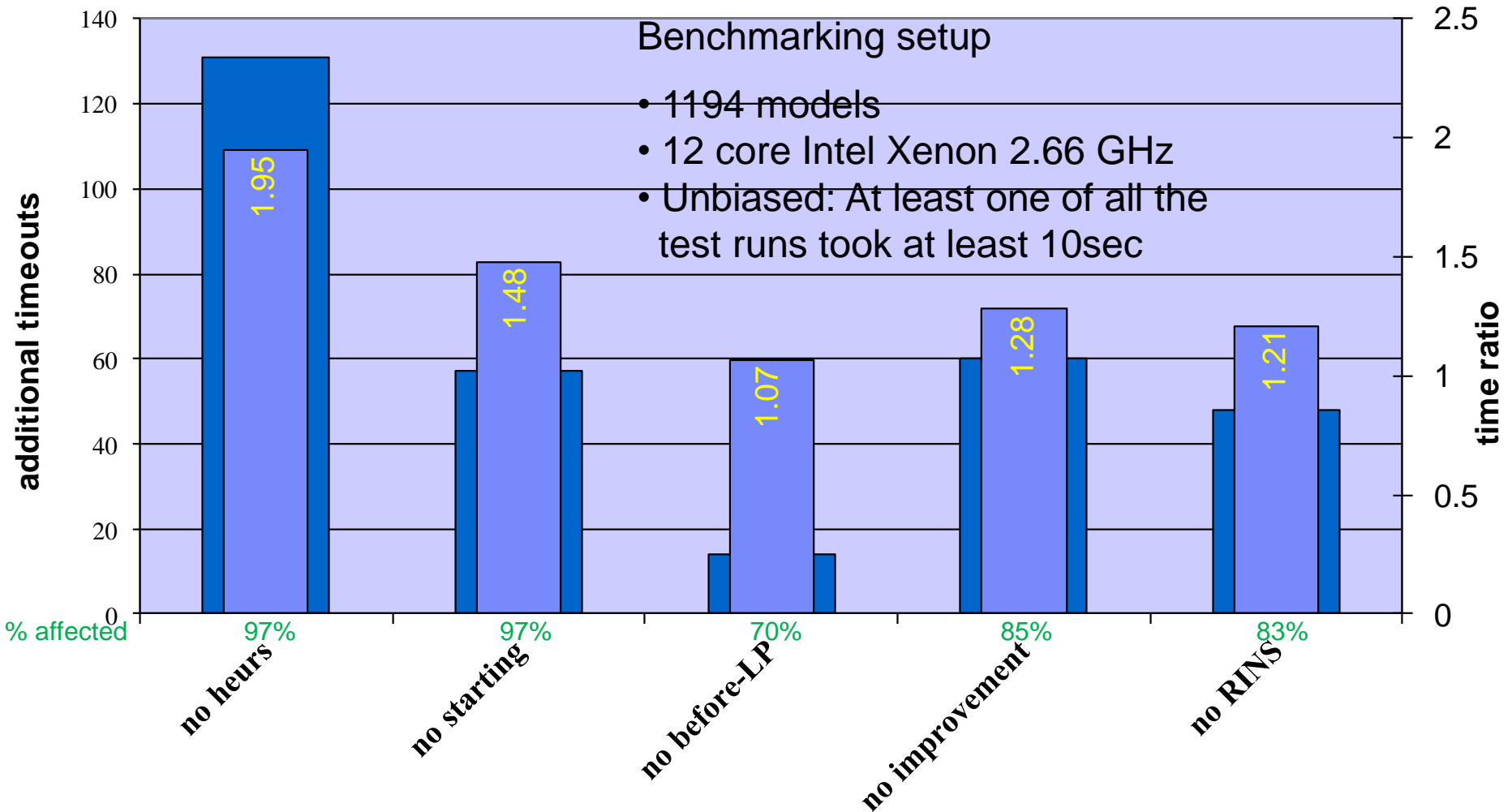
Component Impact CPLEX 12.5.0 – Summary



Component Impact CPLEX 12.5.0 – Summary



Component Impact CPLEX 12.5.0 – Primal heuristics



Legal Disclaimer

- © IBM Corporation 2018. All Rights Reserved.
- The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.
- References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.
- Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.
- Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- Other company, product, or service names may be trademarks or service marks of others.

IBM®