# Efficient Synthesis of Production Schedules by Optimization of Timed Automata

**Inga Krause**

*Institute of Automatic Control Engineering*
*Technische Universität München*

**`inga.krause@mytum.de`**

# Content

- Motivation
- Scheduling
- Timed Automata
- Mixed Integer Linear Programming
- Algorithm of the Optimized Timed Automata
- Search Heuristics & Reduction of the Search Tree
- Experiments & Results
- Evaluation & Summary

# Motivation

- Production can follow different paths
  → Efficient operations in manufacturing or processing systems

- Allocation of resources to production steps (operations)
  → Profit is maximized / costs minimized
  (in the following algorithm: minimizing the makespan)

- Conflicts arising from exclusive resource usage

- Application domains: telecommunications systems, real-time operation systems, robot assembly, chemical industry …

# Job-shop Scheduling

- Scheduling
  - Set of operations “O”, resources “R” and jobs “J”
  - A mapping for the duration of each operation & the assignment of operations to jobs
  - A schedule which assigns a start time to each operation

- Job-shop scheduling as a limitation of scheduling
  - Covering and Non-Preemption
  - Mutual exclusion
  - Operations ordered within a job

- Limitation because efficient algorithms for medium- to large-size problems still do not exist

  →But solution very challenging & extendable for other classes

# Methods to Schedule Operations

- Constraint programming
- Genetic algorithms
- Mathematical programming, e.g. mixed-integer linear programming (MILP) →Software: Cplex
- Reachability algorithms for models given as timed automata (TA) →Software: Kronos, IF, Uppaal, etc.

# Methods to Schedule Operations

- Constraint programming
- Genetic algorithms
- Mathematical programming, e.g. mixed-integer linear programming (MILP) →Software: Cplex
- Reachability algorithms for models given as timed automata (TA) →Software: Kronos, IF, Uppaal, etc.

# Methods to Schedule Operations

- Constraint programming
- Genetic algorithms
- Mathematical programming, e.g. mixed-integer linear programming (MILP) →Software: Cplex
- Reachability algorithms for models given as timed automata (TA) →Software: Kronos, IF, Uppaal, etc.

↓

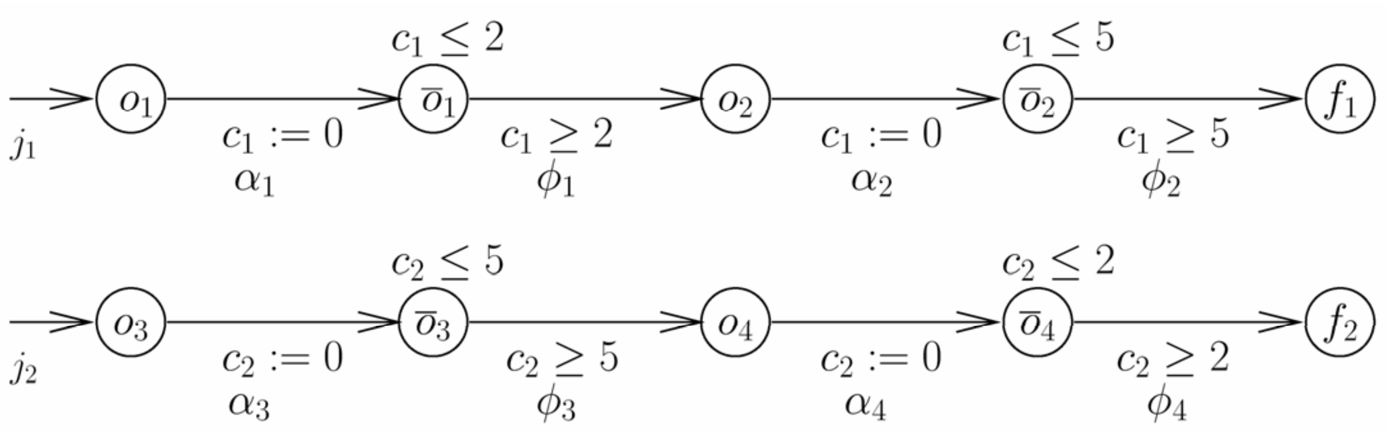
Optimization of timed automata: combination of MILP and TA

# Methods to Schedule Operations

- Optimization of timed automata
  - Combination of MILP and TA
  - Reduction of the scheduling complexity by:
    - The embedded MILP is updated iteratively
    - Extends the notion of non-lazy execution
  - Software TAopt

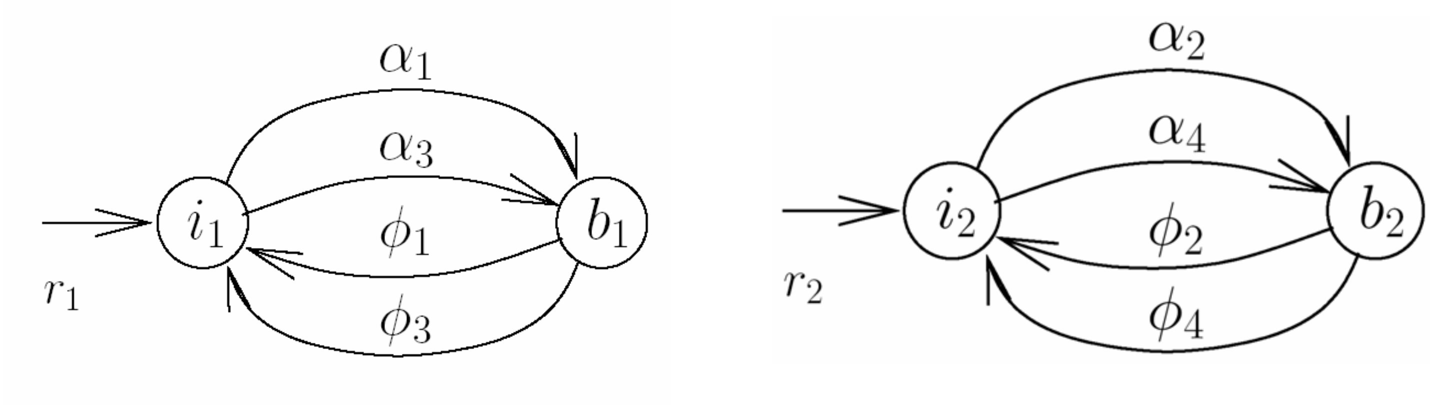
# Timed Automata

- Job-shop Scheduling modeled by TA (makespan minimization)
  - Job automata
    - nodes/locations for each operation
    - operation j is waiting for the resource (=$o_j$) or occupying it (=$\bar{o}_j$)
    - Final location for job i (=$f_i$)
    - Clocks $c_i$ monitor time to occupy a resource
    - Transitions starting an operation and finishing it (labeled with α and Φ)



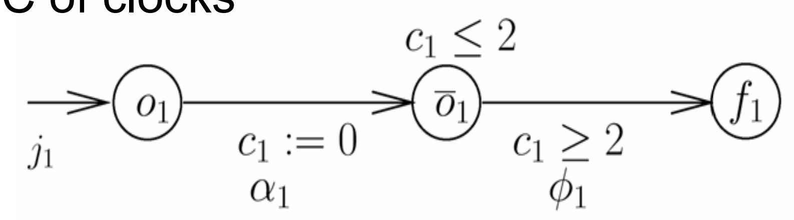
# Timed Automata

- Job Scheduling modeled by TA
  - Resource automatas to enable mutual exclusion
    - For each resource
    - Locations: idle and busy
    - Synchronisations with labels α and Φ



# Timed Automata

- TA is defined as a tuple (L, $l_0$, E, inv)
  - L: finite set of locations (including initial location $l_0$)
  - Inv: L →B(C) "Invariants"
  - E ⊆ L x B(C) x Act x P(C) x L "Transitions"
    - B(C): constraints formulated for a set C of clocks
    - Act: set of actions/labels ($a_i$)
    - P(C): set of reset assignments



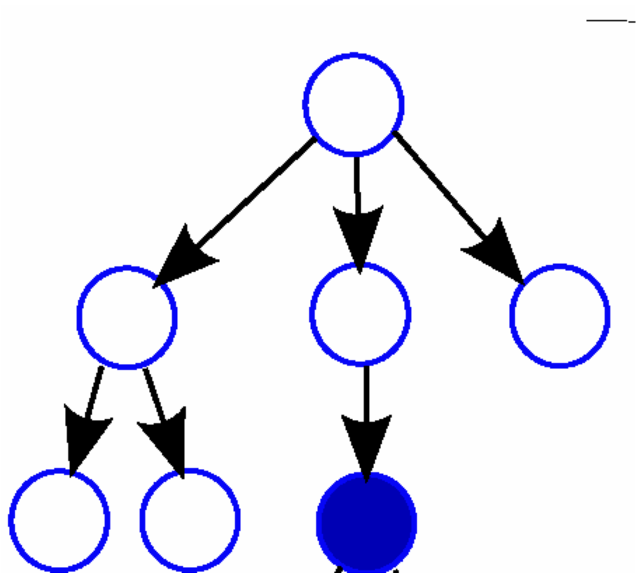

- The semantics of a timed automaton are defined as a labeled transition system (Q,($l_0$,$u_0$),Δ) consists of the state space Q with pairs (l,u) and a transition relation Δ

- A trace of an automaton = sequence of states and transitions

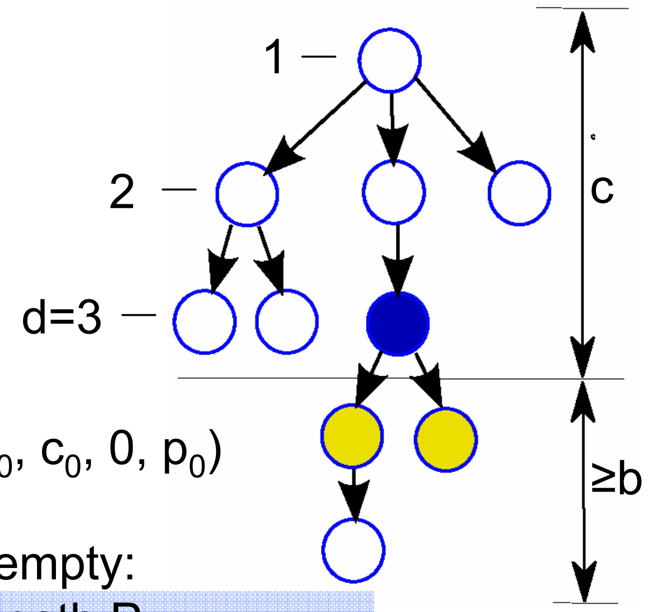
$$\rho = (l_0, u_0) \xrightarrow{a_1} (l_1, u_1) \xrightarrow{a_2} \ldots$$

# Timed Automata

- Reachability Analysis
  - Tree encodes reachable states symbolically as pairs (l,Z) with location l and set Z of clock valuation in l
  - Determine logic properties of TA by exploring reachability tree
  - Extension including cost
    - find path which minimizes cost
    - Software tools: Uppaal CORA, IF, TAopt



# Algorithm of the Optimized Timed Automata




I. Solve relaxed MILP model M with starting node and return b, x

II. Calculates priority p for a node with (b, c, d, x)
- b = lower bound
- c = cost
- d = depth of current node
- x = values of the relaxed decision variables

III. Waiting list W consists of starting node (l, u, b, c, d, p) = ($l_0$, 0, $b_0$, $c_0$, 0, $p_0$)

Now repeat until search tree exploration has been finished and W is empty:

1. Select node from W after chosen heuristic rule and add it to the path P
2. From selected node determine all possible successors S in the TA model A
3. Drop successors with “laziness”
4. Drop already visited nodes

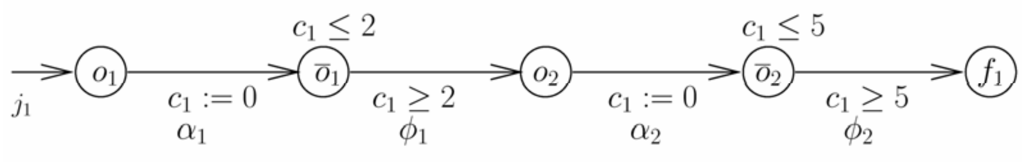
   Now repeat with the remaining nodes in S until every node has been explored (determine new upper bound c* when arriving the target location l*)
   a. Update the model M (past nodes and transitions are fixed now)
   b. Solve relaxed MILP with the updated model (→ b’, x’)
   c. Calculate priority p’ of node (b’, c’, d’, x’)
   d. Add node (l’, u’, b’, c’, d’, p’) to the list W and take next node of S
5. S is empty

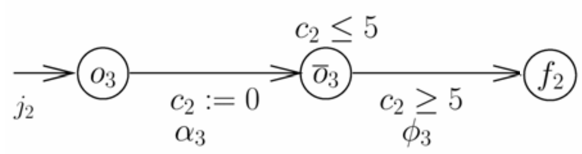
# Algebraic Mixed-Integer Linear Programming Formulation

- Formulation
  - binary variables: sequences of operations on the same resource (with p(o,o')=1 if operation o should be started first, 0 otherwise)
  - Continuous variables: starting dates of operations, duration of the operations, time horizon H
  - Equations for execution order, mutual exclusion, time horizon
  - Inequalities for all operations to minimize problem (makespan)

Example:

Time horizon: H=100
Execution order: $s(o_1)+2 \leq s(o_2)$
Mutual exclusion: $p(o_1,o_3) + p(o_3, o_1) = 1$
Makespan: min $\Psi$ with
$s(\bar{o}_2) + d(\bar{o}_2) \geq \Psi$, $s(\bar{o}_3)+d(\bar{o}_2) \geq \Psi$'



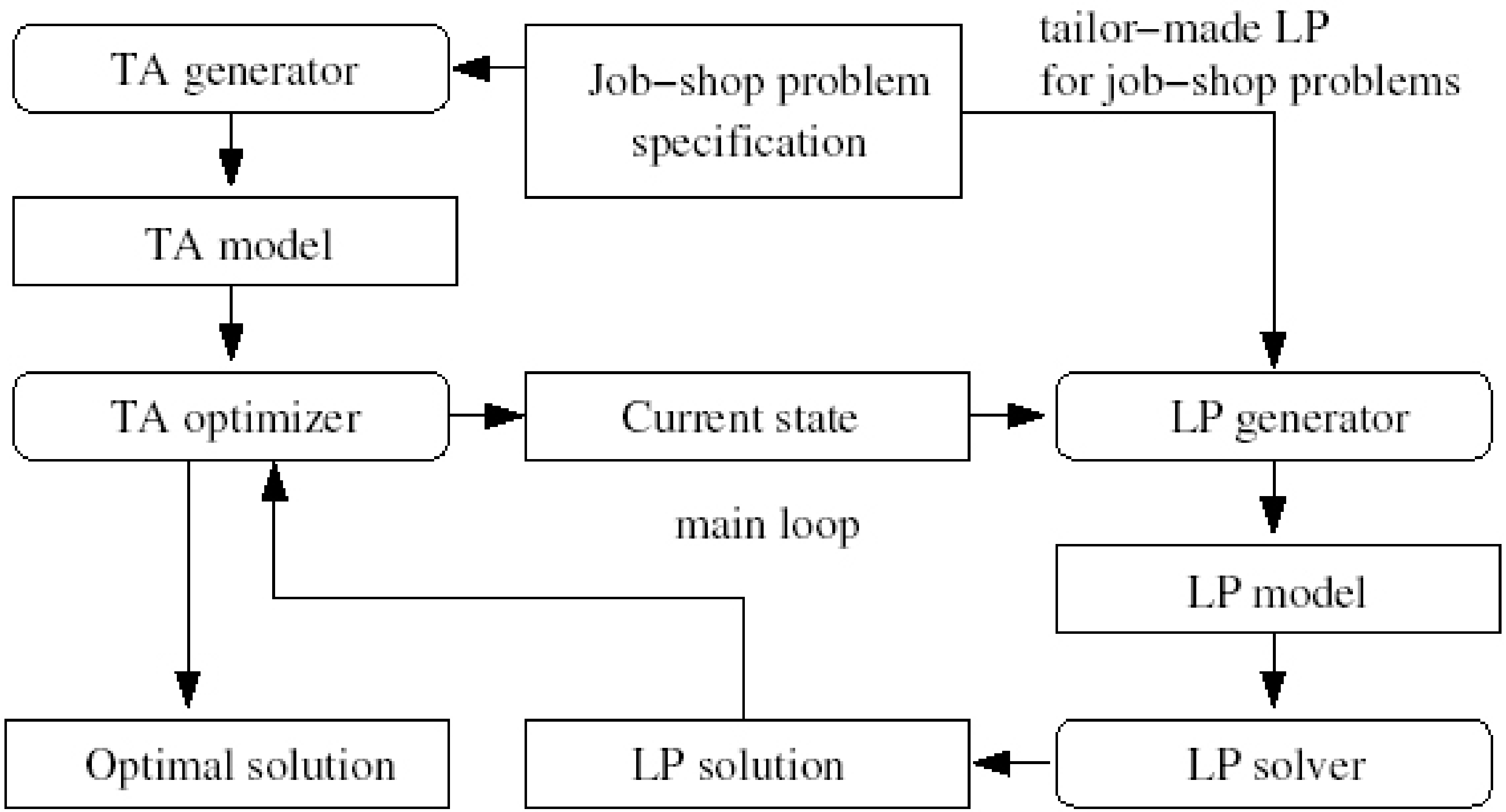



# Algebraic Mixed-Integer Linear Programming Formulation

- Result of solved MILP model with current state:
  - Solution vector x of relaxed decision variables
  - Lower bound to prune reachability tree

# Timed Automata with embedded LP

- Structure of the implementation



# Search Heuristics

- Selection criteria for next node
- Based on node attributes (b, c, d, p)
- Influence the search tree exploration towards the optimum
- Determine the performance of finding the optimum
- Combination of selection criteria often useful

| Depth-First Search or Breadth-First Search | Min-Cost Search (Best-First Search) | Best-Lower-Bound Search | Random Search | Compute Priority |
|---|---|---|---|---|
| Select node with maximal/minimal depth d | Select node with minimal cost c | Select node with lowest bound b | Select on random distributed priorities p | Select node with highest priority p (evaluation of LP-solution vector x) |

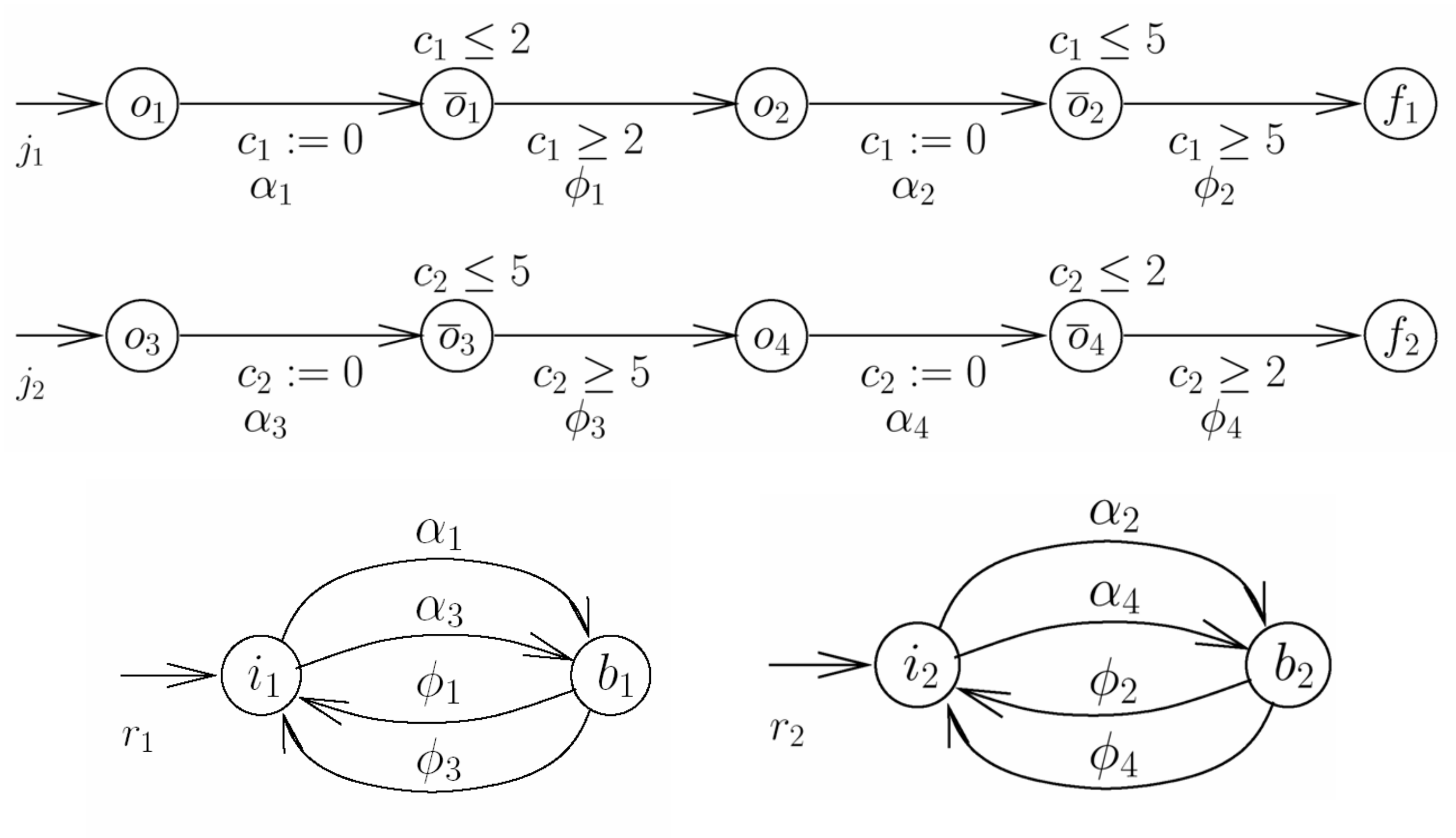
# Search Heuristics

- Selection criteria for next node
- Based on node attributes (b, c, d, p)
- Influence the search tree exploration towards the optimum
- Determine the performance of finding the optimum
- Combination of selection criteria often useful

| Depth-First Search or Breadth-First Search | Min-Cost Search (Best-First Search) | Best-Lower-Bound Search | Random Search | Compute Priority |
|---|---|---|---|---|
| Select node with maximal/minimal depth d | Select node with minimal cost c | Select node with lowest bound b | Select on random distributed priorities p | Select node with highest priority p<br>(evaluation of LP-solution vector x) |

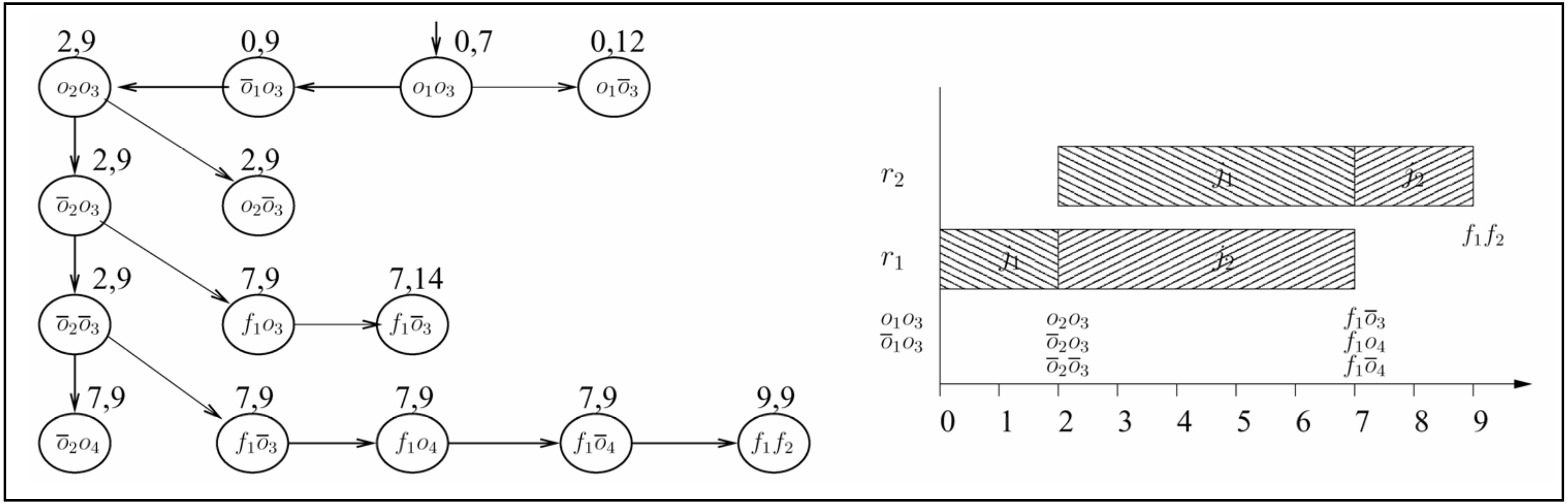
# Search Heuristics Example (1)

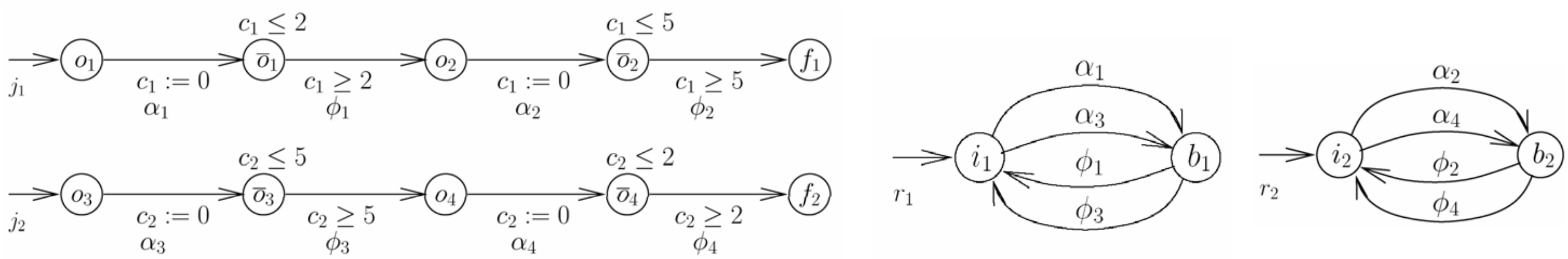
- Best-lower-bound heuristics to find optimal path from $o_1o_3$ to $f_1f_2$



# Search Heuristics Example (2)

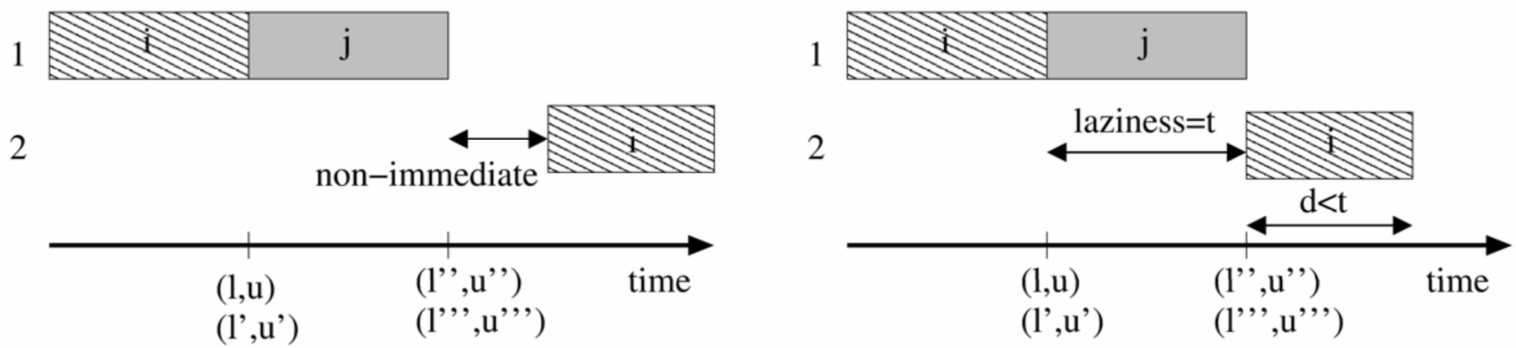
- Nodes are represented by location of both job automata $l_1$ and $l_2$, accumulated cost c and lower bound: $(l_1, l_2, c, b)$



# Reduction of the Search Tree

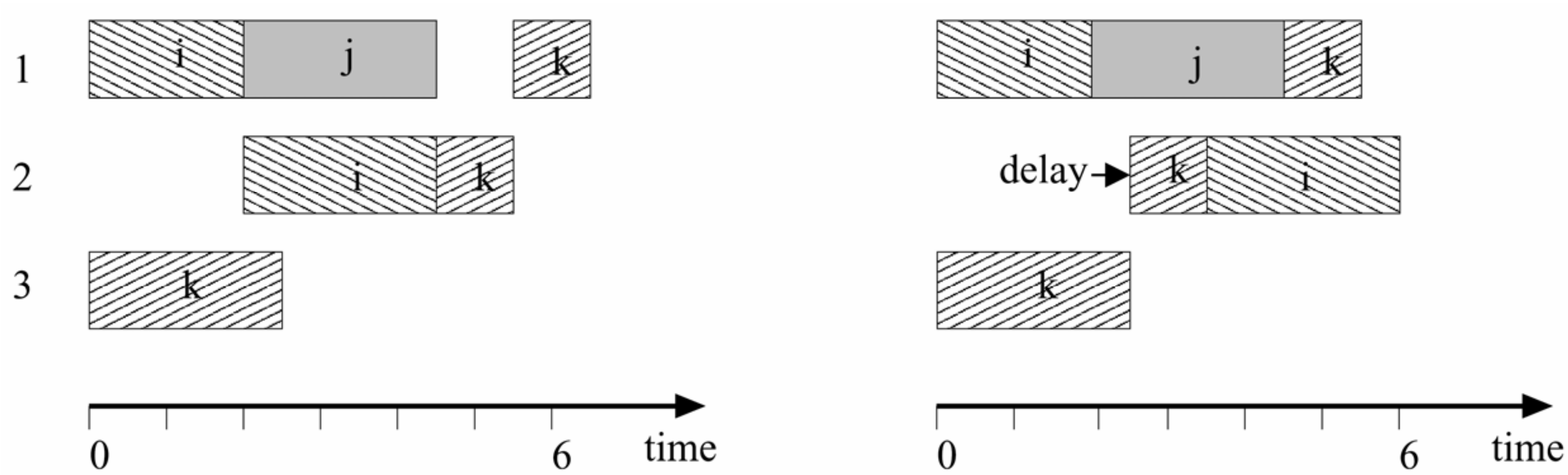
- Properties of traces:
  - Immediate
    - Traces do not exhibit periods of useless waiting (no task is started although resources available)
    - Optimal trace always immediate in the case of makespan and tardiness minimization
    - For general cost functions waiting before/after operations can be advantageous




  - Weak non-lazy
    - Stricter criterion than immediate
    - Time gaps/holes which are large enough to be filled with an enabled operation are forbidden

# Reduction of the Search Tree

- Properties of traces:
  - Strong non-lazy / greedy strategy
    - Even more restrictive: whenever list of successor states is not empty, waiting is forbidden in the current state (l,u)
    - If no new operations can be started
    - Not always optimal (see figure: left strongly non-lazy, right weakly non-lazy)



# Experiments (1)

- Two kind of test series:
  - Series A:
    - randomly small instance with no. of operation = no. of resources for each job
    - Duration of operation uniformly distributed over {1, 2, …,6}
    - Operations randomly assigned to the resources for each job
  - Series B:
    - Set of job-shop benchmark instances

# Experiments (2)

- Software used:
  - TAopt (with Cplex to solve the embedded LP problem)
    - cost-optimal reachability algorithm for TA employing branch-and-bound
    - computation of lower bounds from embedded LP problems
    - various node selection criteria (like Depth-First Search etc.)
    - weak and strong non-laziness
  - Cplex (pure MILP solving)
  - Kronos (pure TA without lower costs)

# Results with series A (1)

- Number of explored nodes for
  - different no. of jobs and resources
  - Different search tree reduction techniques
    - branch-and-bound techniques with embedded LP
    - weak or strong non-laziness

bb = Branch and bound

wnl = weak-non-laziness

snl = strong non-laziness

| Jobs/ Res | pure | bb | wnl | snl |
|---|---|---|---|---|
| 2/2 | 31 | 12 | 27 | 25 |
| 3/2 | 214 | 87 | 218 | 95 |
| 4/2 | 1074 | 189 | 641 | 233 |
| 5/2 | 6343 | 2367 | 3155 | 1042 |
| 6/2 | 37553 | 15949 | 46943 | 5803 |
| 2/3 | 62 | 19 | 38 | 21 |
| 3/3 | 741 | 142 | 284 | 73 |
| 4/3 | 7445 | 1631 | 1165 | 366 |
| 5/3 | 87365 | 13843 | 34772 | 3392 |
| 6/3 | 549324 | 65934 | 316120 | 1182 |
| 2/4 | 126 | 26 | 72 | 39 |
| 3/4 | 2631 | 582 | 722 | 103* |
| 4/4 | 35172 | 4677 | 5189 | 641 |
| 5/4 | 451290 | 28394 | 50163 | 3004 |
| 6/4 | 5253234 | 1032669 | 738293 | 9503* |

# Results with series A (2)

- Heuristics
  - min c is bad choice
  - Combined criteria are preferable, in particular the combination of min b & max d

| Jobs/Res | (a) | (b) | (c) | (d) | (e) | (f) | (g) | (h) | (i) |
|---|---|---|---|---|---|---|---|---|---|
| 2/2 | 21 | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 12 |
| 3/2 | 96 | 42 | 42 | 42 | 42 | 58 | 83 | 42 | 42 |
| 4/2 | 211 | 89 | 89 | **53** | **53** | 61 | 98 | **40** | **40** |
| 5/2 | 843 | 415 | 415 | **386** | **386** | 422 | 519 | **378** | **378** |
| 6/2 | 4810 | 2562 | 2562 | **1058** | **1058** | 1202 | 1983 | **1061** | **1047** |
| 2/3 | 19 | 19 | 19 | 19 | 19 | 19 | 19 | 19 | 19 |
| 3/3 | 54 | 58 | 58 | 58 | 58 | 47 | 47 | 47 | 47 |
| 4/3 | 282 | 143 | 143 | **114** | **114** | 147 | 158 | **114** | **114** |
| 5/3 | 2417 | 831 | 831 | **648** | **648** | 1038 | 1177 | **508** | **508** |
| 6/3 | 12640 | **1906** | **1906** | **1851** | **1851** | 2677 | 3511 | 2118 | 2070 |
| 2/4 | 27 | 27 | 27 | 27 | 27 | 27 | 27 | 27 | 27 |
| 3/4 | 75 | 75 | 75 | 42 | 42 | 42 | 42 | 42 | 42 |
| 4/4 | 410 | 232 | 232 | **99** | **99** | 151 | 168 | **99** | **99** |
| 5/4 | 1628 | 638 | 638 | **446** | **446** | 493 | 721 | **434** | **434** |
| 6/4 | 6379 | 4045 | 4045 | **3919** | **3919** | 4079 | 6051 | **3817** | **3794** |

Search strategies:
(c=cost, b=lower bound,
c=upper bound, d=depth)
(a) = min c
(b) = max d
(c) = max d, min c
(d) = max d, min b
(e) = max d, min c
(f) = min b
(g) = min b, min c
(h) = min b, max d
(i) = min b, max d, min c

# Results with series A (2)

- Heuristics
  - min c is bad choice
  - Combined criteria are preferable, in particular the combination of min b & max d

Search strategies:
(c=cost, b=lower bound,
c=upper bound, d=depth)
(a) = min c
(b) = max d
(c) = max d, min c
(d) = max d, min b
(e) = max d, min c
(f) = min b
(g) = min b, min c
(h) = min b, max d
(i) = min b, max d, min c

| Jobs/Res | (a) | (b) | (c) | (d) | (e) | (f) | (g) | (h) | (i) |
|---|---|---|---|---|---|---|---|---|---|
| 2/2 | 21 | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 12 |
| 3/2 | 96 | 42 | 42 | 42 | 42 | 58 | 83 | 42 | 42 |
| 4/2 | 211 | 89 | 89 | **53** | **53** | 61 | 98 | **40** | **40** |
| 5/2 | 843 | 415 | 415 | **386** | **386** | 422 | 519 | **378** | **378** |
| 6/2 | 4810 | 2562 | 2562 | **1058** | **1058** | 1202 | 1983 | **1061** | **1047** |
| 2/3 | 19 | 19 | 19 | 19 | 19 | 19 | 19 | 19 | 19 |
| 3/3 | 54 | 58 | 58 | 58 | 58 | 47 | 47 | 47 | 47 |
| 4/3 | 282 | 143 | 143 | **114** | **114** | 147 | 158 | **114** | **114** |
| 5/3 | 2417 | 831 | 831 | **648** | **648** | 1038 | 1177 | **508** | **508** |
| 6/3 | 12640 | **1906** | **1906** | **1851** | **1851** | 2677 | 3511 | 2118 | 2070 |
| 2/4 | 27 | 27 | 27 | 27 | 27 | 27 | 27 | 27 | 27 |
| 3/4 | 75 | 75 | 75 | 42 | 42 | 42 | 42 | 42 | 42 |
| 4/4 | 410 | 232 | 232 | **99** | **99** | 151 | 168 | **99** | **99** |
| 5/4 | 1628 | 638 | 638 | **446** | **446** | 493 | 721 | **434** | **434** |
| 6/4 | 6379 | 4045 | 4045 | **3919** | **3919** | 4079 | 6051 | **3817** | **3794** |

# Results with series B

- Comparison of software in finding the optimum within 1 hour computation time
  - Cplex: the best for small instances
  - TAopt: the best for big instances
  - Kronos: no solution for small instances, half of the time better solutions than TAopt

- Comparison of upper bounds using TAopt and Cplex
  - TAopt: slightly more time to find a first feasible solution than Cplex
  - TAopt: significantly lower upper bounds

# Evaluation

- Extension to more general scheduling problems possible:
  - Alternative production paths
  - Parallel production paths
  - Timing constraints
  - Changeover procedures
  - Restricted working times
  - Size of material stocks
  - Consumption of material and production of another material

# Summary

- Recent approach of combining reachability analysis of TAs with branch-and-bound principle and non-lazy traces
- Results of TA for several samples even better than pure MILP
- Problem formulation for TA-based methods more intuitive
- Benefits of minimizing search larger than the increase of additional computation time per node
- Smaller memory consumption through pruning

# References


- S. Panek, S. Engell, and O. Stursberg. Scheduling and Planning with Timed Automata. In *16th Europ. Symp. on Computer-Aided Process Engineering*, pages 1973-1978, 2006
- S. Panek, O. Stursberg, and S. Engell. Efficient Synthesis of Production Schedules by Optimization of Timed Automata. *Control Engineering Practice*, 14(10):1183-1197, 2006
- S. Panek, O. Stursberg, and S. Engell. Optimization of Timed Automata Models using Mixed-Integer Programming. In *Formal Modeling And Analysis of Timed Systems*, volume 2791 of LNCS, pages 73-87. Springer, 2004.
- S. Panek, O. Stursberg, and S. Engell. Job-Shop Scheduling by Combining Reachability Analysis with Linear Programming. In *7th Int. IFAC Workshop on Discrete Event Systems*, page 199-204, 2004
- J. Kallrath. Combined strategic and operational planning – an MILP success story in chem. Ind.. *OR Spectrum* 24, pages 315-241, 2002
- Y. Abdeddaim and O. Maler. Job-shop scheduling using timed automata in *Computer Aided Verification (CAV)*. volume 2102 of LNCS, pages 478-492, Springer , 2001
- E. Kondili, C. Pantelides, and R. Sargent. A general algorithm for short-term sched. of batch operations – MILP formulation. *Comp. Chem. Eng*. 17, pages 211-227, 1993