Parpap: Scheduling of Ambulant Nursing Staff

– Constraint Programming based Column Generation –

Torsten Fahle,

University of Paderborn / Germany

tef@upb.de

Partners in the PARPAP project:

VSS Bremen University of Karlsruhe University of Paderborn

3 Health Care Organizations

Project PARPAP supported by German Ministry for Education and Research (bmb+f)

Content

The Optimization Problem in Parpap

Assign n nurses/vehicles to m patients/locations such that the solution

- satisfies all legality, qualification and working hours constraints,
- mets as many as possible preferences of patients and nurses,
- follows as many as possible recommendations from ergonomics.

objective maximize patients'/nurses' satisfaction, minimize costs

$\Rightarrow$ Vehicle Routing Problems with additional constraints

$\Rightarrow$ Crew Rostering with Preferential Bidding

Soft and Hard Constraints for Nurse Scheduling

Preferences of Patients

- always the same nurse
- male/female/nationality
- always the same time
- time for talking ,
- . . .

Preferences of Nurses

- days off,
- late start (children)
- weekend shifts
- . . .

Legal Aspects

- working time,
- right qualification,
- different service times
- . . .

Ambulant Nurse Scheduling

Qualification/Experience

- Nurse/civil service/. . .
- over qualified ?
- special qualifications
- . . .

Ergonomics

- light/heavy patients
- shift plans
- ability to take stress
- . . .

others . . .

- vehicle types
- traffic situation
- contracts
- right 'chemistry'
- . . .

Literature Review

Ambulant Nurse Scheduling

Cheng, Rich A Home Health Care Routing and Scheduling Problem, 1998.

Hospital Nurse Scheduling

Abdennadher, Schlenker Nurse Scheduling using Constraint Logic Programming, 1999

Burke et al. A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem, 1999.

Mason, Smith A Nested Column Generator for solving Rostering Problems with Integer Programming, 1998.

Cheng, Lee, Wu A Nurse Rostering System Using Constraint Programming and Redundant Modeling, 1997.

. . .

Vehicle Routing . . .

Generic Approach in Parpap

- one software to be designed for three different health care organizations
  - single depot / multi depot
  - working on demand / contract on working time
  - 1 day planning period / 2 weeks planning period
  - ...
- Laws and regulations change quickly
  - 5 major changes in the last 15 month in Germany
  - often more than just 'parameter adjustment'

⇒ There is a need for a flexible modeling that allows also nonlinear constraints, etc.

⇒ Constraint Programming based Column Generation [Junker et al 1999] {Ulrich Junker, Niklas Kohl, Stefan Karisch, Bo Vaaben, Meinolf Sellmann, T. Fahle}

Constraint Programming – A Quick Reminder

- With each variable $x \in \mathcal{X}$ a finite domain $D(x)$ is associated,
- Constraints $C \in \mathcal{C}$ couple one or more variables $C(x_1, \ldots, x_k)$
- Propagation removes inconsistent values from the set of possible variable assignments.
- As long as the domains are not unique after propagation branch into 2 subproblems (and propagate again)

Example: Constraint Programming for Nurse Scheduling

- 2 nurses N_1 (q_1, q_2), N_2 (q_1)
- 4 patients P_1 $(9:10, q_1)$, P_2 $(11:12, q_1)$, P_3 $(9:10, q_2)$, P_2 $(12:13, q_1)$,
- (R_1) only one patient per time, (R_2) qualification must fit! (R_3) two patients per nurse

initial $pos(N_1) = pos(N_2) = \{P_1, P_2, P_3, P_4\}, \quad req(N_1) = req(N_2) = \emptyset$

(R_2) $req(N_1) = \{P_3\}$, $pos(N_2) = \{P_1, \cancel{P}_3, P_2, P_4\}$

(R_1) $req(N_2) = \{P_1\}$, $pos(N_1) = \{\cancel{P}_1, P_2, P_3, P_4\}$

branching on $\boldsymbol{P_3}$ $req(N_1) := req(N_1) \cup \{P_3\}$

$\rightarrow$ $pos(N_2) = \{\cancel{P}_1, P_2, \cancel{P}_3, P_4\}$

(R_3) $pos(N_1) = \{P_1, \cancel{P}_3, P_2, \cancel{P}_4\}$

(R_3) $pos(N_2) = \{\cancel{P}_1, P_2, \cancel{P}_3, P_4\}$, $req(N_2) := \{P_2, P_4\}$

solution found

Traditionally: Constraint Programming or Column Generation

Column Generation

$\boxed{\text{Constr. Shortest Path}}$

duals $\uparrow$ $\downarrow$ columns

$\boxed{\text{LP solver}}$

- limited expressiveness,
- but powerful optimization

CSP

$\boxed{\begin{array}{c}\text{CSP}\\ \text{(propagation)}\end{array}}$

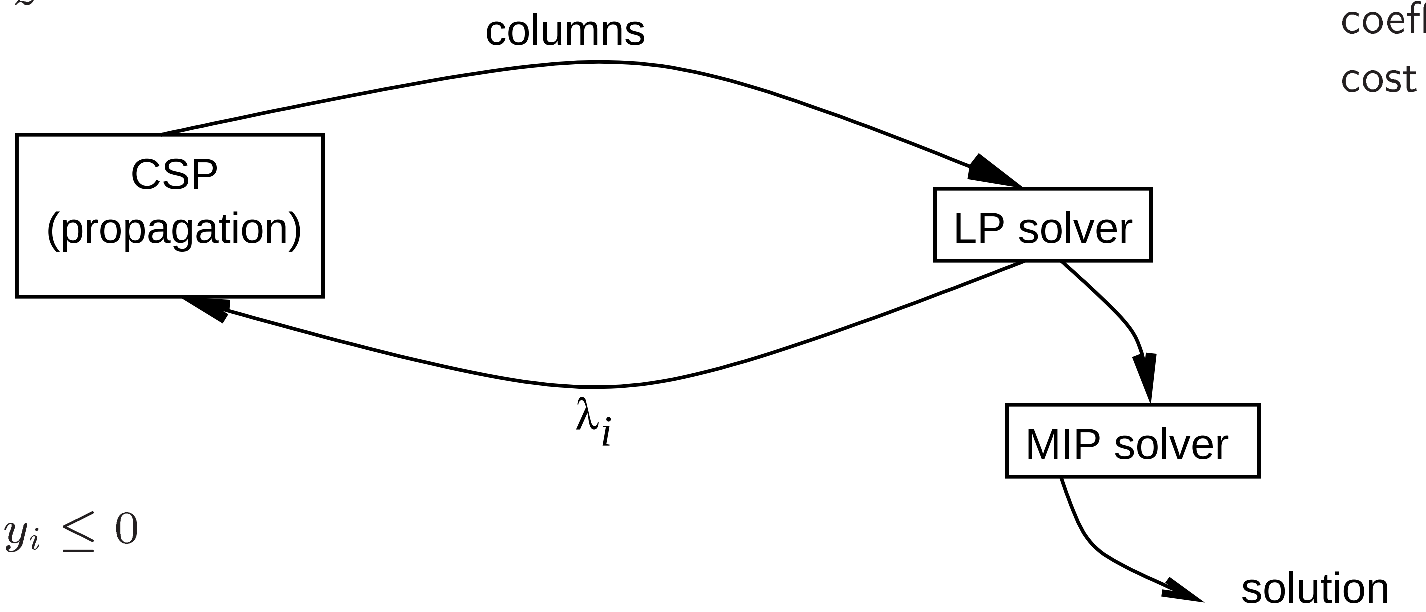
- high expressiveness,
- powerful propagation,
- but limited optimization

Constraint Programming based Column Generation

Combine both ideas!

solution y_i, z
($=$ path)

Variable x_j,
coefficient $a_{i,j}$,
cost c_j



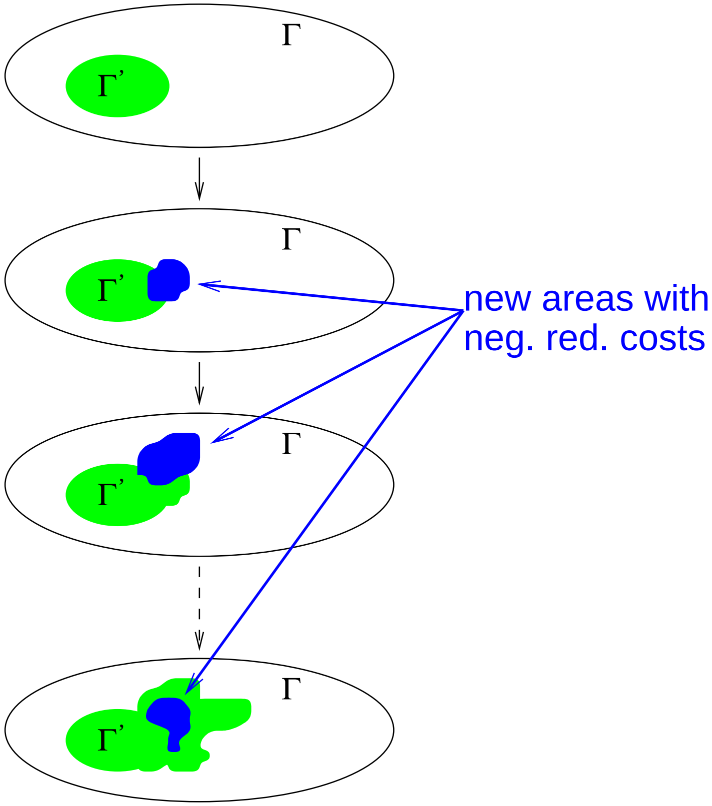
$$z - \sum_{i \in \mathcal{N}} \lambda_i y_i \leq 0$$

(NRC constraint)

Outline of Column Generation

Subproblem:

- Generate further columns with negative reduced costs
- Use dual values of LP solution to determine reduced costs



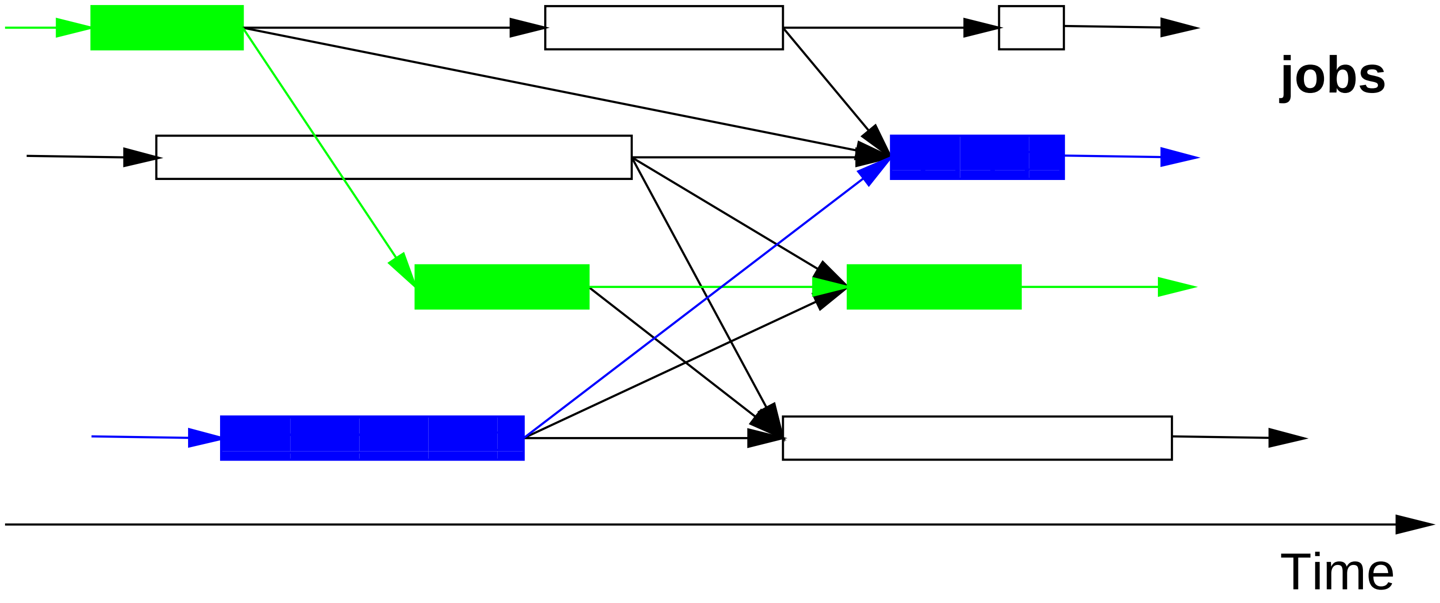
Γ' :=`getInitialColumns()`
repeat
  $\lambda :=$ `solveLP`(Γ') {get new duals}
  $\{x_{j_1}, \ldots, x_{j_k}\} :=$ `solveSubproblem`(λ)
  $\Gamma' = \Gamma' \cup \{x_{j_1}, \ldots, x_{j_k}\}$
until $(\{x_{j_1}, \ldots, x_{j_k}\} = \emptyset)$

Constraint Shortest Paths

Staff Rostering

- Assign rosters to named individuals
- respect many rules and regulations

} shortest path (= lower bound for feasible roster)



The Subproblem: Generate a single 'good' path

Modify Network: Add source s, sink t, edges (s, i), (i, t) to DAG $\rightarrow \mathcal{N}$

Subproblem: Find an (s, t)-path with negative reduced costs

$y_i \in \{0, 1\}$	$y_i = 1 \iff$ node i is on that path
z	cost of that path

For the simple problem: $y_s = 1$, $y_t = 1$, $z = 1$

negative reduced costs constraint (NRC)

$$\boxed{z - \sum_{i \in V} \lambda_i y_i \leq 0}$$

(where λ_i is the dual value of the constraint corresponding to node i in the master problem)

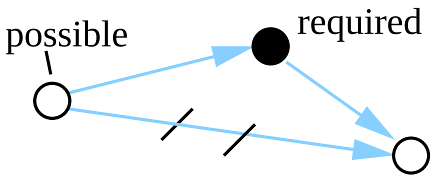
Path constraint

- directed acyclic graph $\mathcal{N} = (V, E)$, sorted topologically, source s, sink t
- edge costs $c_{i,j}$
- constrained set variable $Y \subseteq \mathcal{N}$
- constrained variable z (path costs)

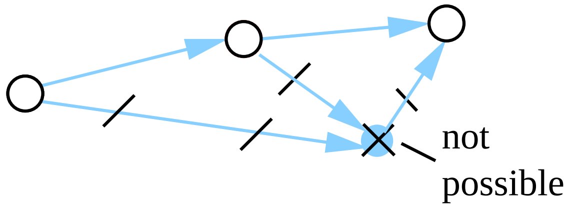
1. Y represents a path $\{(i, \text{next}(i, Y)) \mid i \in Y \cup \{s\}\} \subseteq \mathcal{N}$
2. z is sum of edge costs $z = \sum_{i \in Y \cup \{s\}} c_{i,\text{next}(i,Y)}$

(next$(i, Y) := \min\{j \in Y \cup \{t\} \mid j > i\}$ is the direct successor of node i on the path Y)

Basic Propagation



- bounds: req$(Y) \subseteq Y \subseteq$ pos(Y), $\min(z) \leq z \leq \max(z)$
- remove edges around elements of req(Y)
- consider only nodes in pos(Y)



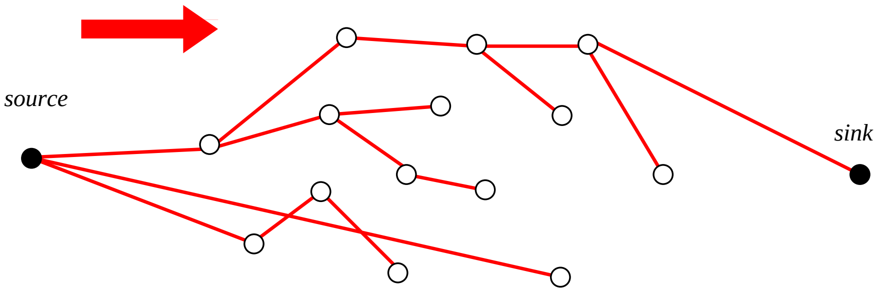
- determine shortest paths from source s to all nodes i: $z_{s,i}$
- determine shortest paths from sink t to all nodes i: $z_{i,t}$

$\Rightarrow$ SSSP algorithm for DAGs $\rightarrow O(|V| + |E|)$

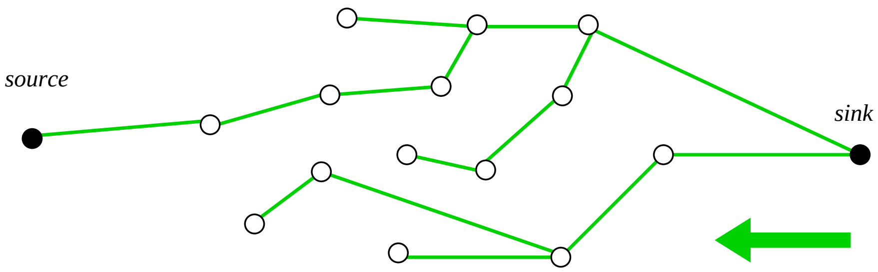
Propagations:

- $\min(z) \geq z_{s,t}$
- $z_{s,i} + z_{i,t} > \max(z) \Rightarrow$ pos$(Y) :=$ pos$(Y) \setminus \{i\}$
- $z_{s,i} + c_{i,j} + z_{j,t} > \max(z) \Rightarrow$ suppress edge (i, j)

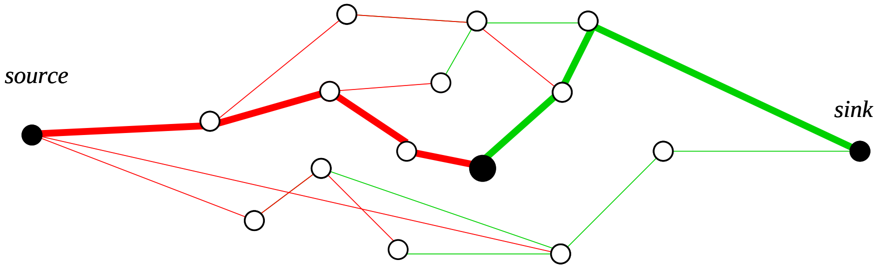
Example



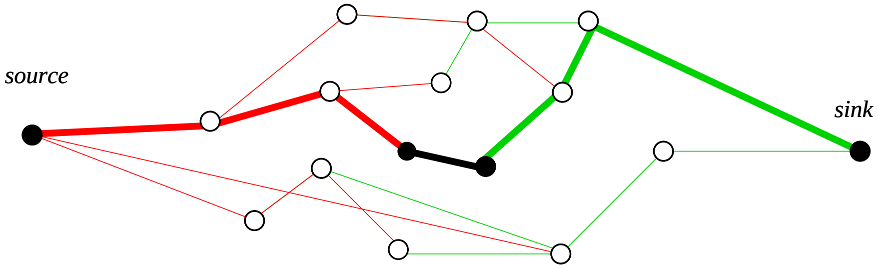
shortest paths from source to all nodes



shortest paths from sink to all nodes



$z_{s,i} + z_{i,t} > \max(z)$
$\Rightarrow \mathsf{pos}(Y) := \mathsf{pos}(Y) \setminus \{i\}$



$z_{s,i} + c_{i,j} + z_{j,t} > \max(z)$
$\Rightarrow$ suppress edge (i, j)

Torsten Fahle
University of Paderborn

Interference with other constraints

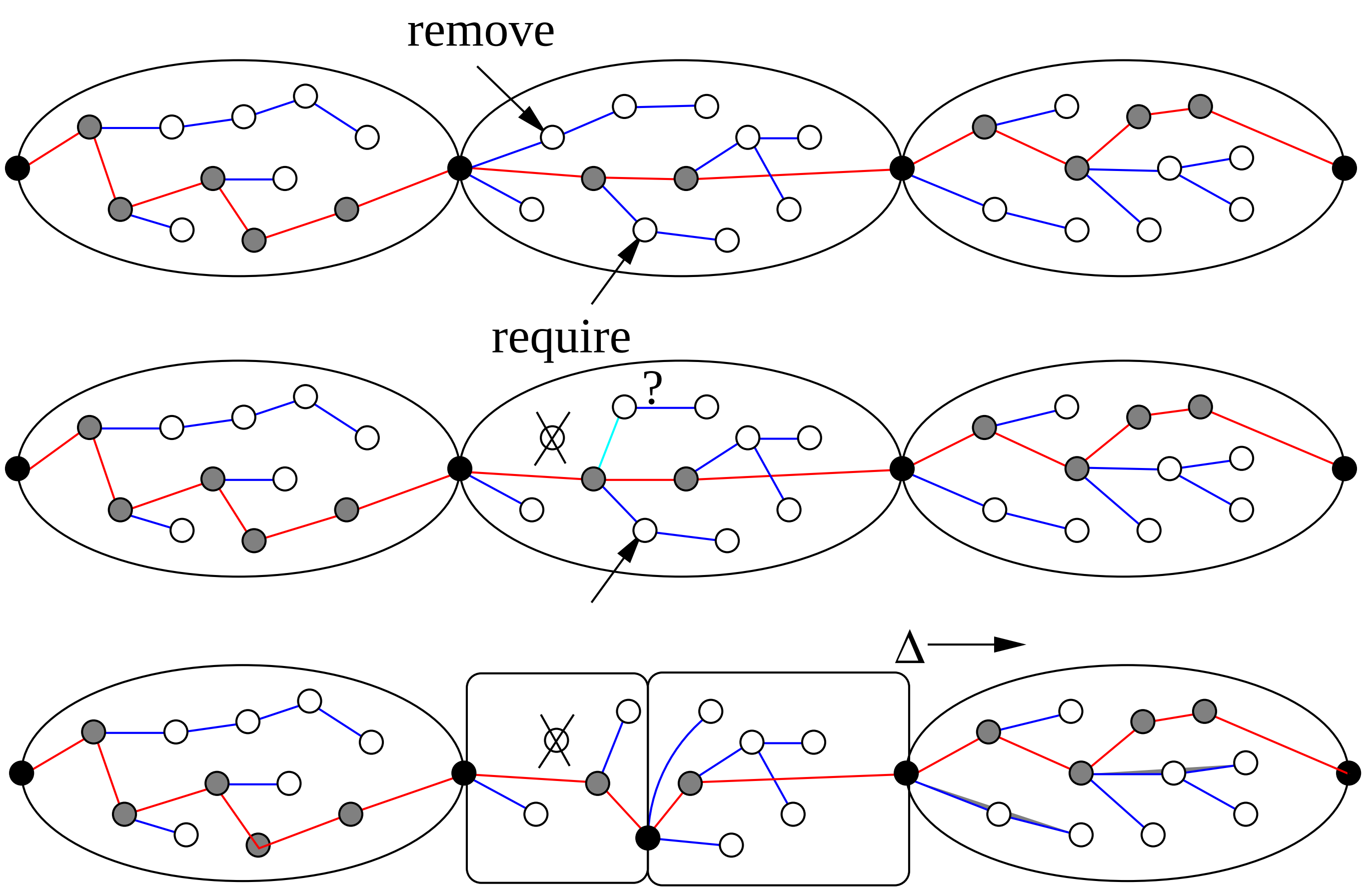
- Each time an edge/node is required it is added to the required set of the path
- This may trigger propagation of other constraints (work time limits, exclude overlapping jobs, . . .).
- This may trigger the shortest path constraint again.

$\Rightarrow$ all constraints communicate via domains.

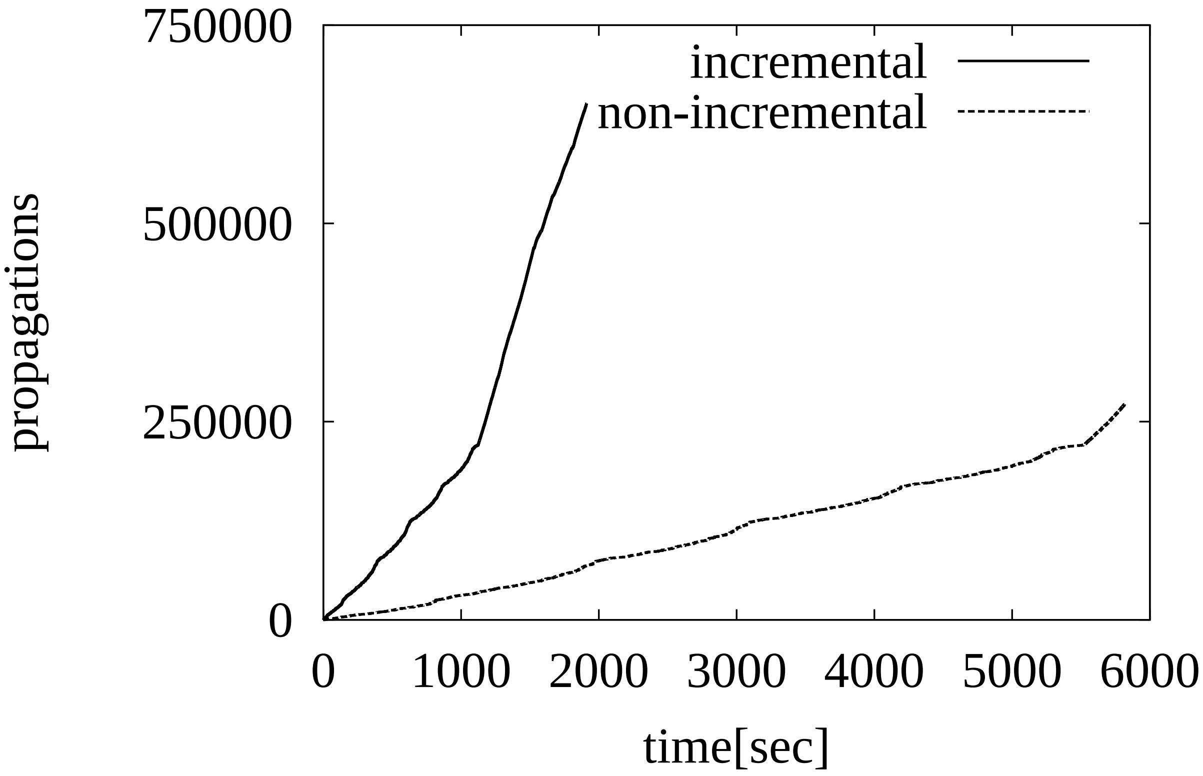
$\Rightarrow$ an incremental shortest path algorithm may save a lot of computational effort.

(similar behaviour if a edge/node is not possible anymore)

Incremental Shortest Path



Influence of Incremental Shortest Path Update



(tested on aircraft crew rostering instances)

Conclusion

- Ambulant Nurse Scheduling combines VRP and staff rostering
- CP used for generic modeling of all constraints
- Column Generation for getting good quality solutions.
- Shortest Path Constraint is common for all problem instances.
  - Efficient incremental propagation algorithm for shortest path on DAGs.
  - Shortest path and all other constraints communicate via shared variables/domains.