n the facial structure of and e cient solution methods for the opportunistic maintenance problem

Magnus Önnheim

December 11, 2009

Opportunistic maintenance problem

- Use corrective maintenance as an opportunity to perform preventive maintenance
- Optimization group at Chalmers has shown that such models can have superior performance compared to simpler policies (Patriksson et al 2009)
- However solution times grows very quickly with the planning horizon in a general solver
- Special algorithms/heuristics required

Basic Replacement Model

- Parameters $n, T, T_i,\ i \in \mathcal{N}$
- Variables $x_{it}, z_t\ i \in \mathcal{N}, t \in \mathcal{T}$
- Costs $c_{it}, d_t, i \in \mathcal{N},\ t \in \mathcal{T}$

Basic Replacement Model

$$\min_{(x,z)} \sum_{t \in \mathcal{T}, i \in \mathcal{N}} c_{it} x_{it} + \sum_{t \in \mathcal{T}} d_t z_t \quad (1a)$$

$$\text{s.t.} \sum_{t=l+1}^{l+T_i} x_{it} \geq 1 \qquad i \in \mathcal{N}, l \in \{0, \ldots, T - T_i\} \quad (1b)$$

$$x_{it} \leq z_t \qquad i \in \mathcal{N}, t \in \mathcal{T} \quad (1c)$$

$$x_{it}, z_t \in \{0, 1\} \qquad i \in \mathcal{N}, t \in \mathcal{T} \quad (1d)$$

Some conditions and consequences

- Assume that costs are non-increasing in t
- Then the following inequality holds for some optimal solution

$$\sum_{i=1}^{n} x_{it-T_i} \geq z_t, \quad t \in \mathcal{T} \quad (2)$$

- Greedy rule if maintenance dates are known: only replace components that fail before the next maintenance occasion
- At least one maintenance is corrective

Dynamic Programming Scheme

- Shortest Path Based in the digraph $\mathcal{D} = (V, E)$, Markov state space

$$V = \{(t, \tau_1, \ldots, \tau_n)\} \cup (0, T_i) \cup (T+1, 0, \ldots, 0),\ t \in \mathcal{T},\ \tau_i \in \{0, \ldots, T_i - 1\} \quad (3)$$

- Each node contains enough information about the system to determine future maintenance
- Greedy rule and (2) yields great *a priori* reductions on the number of edges

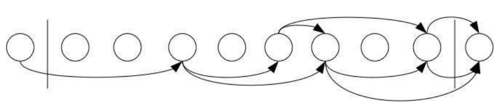
Dynamic Programming Scheme

- How to define edges E from a fixed state $(t, \tau_1, \dots, \tau_n)$?
- Want them to connect to the next maintenance occasion $(t', \tau'_1, \dots, \tau'_n)$
- Given by (2) as $t' = t + \tau_i$ or $t' = t + T_1$, if $\tau_i \le T_1$
- When known, greedy rule determines maintenance at t, yielding τ'_i
- At most n edges from one state
- Additional edges to the extra state $(T+1, 0, \dots, 0)$ if there is a feasible maintenance decision reaching beyond the end of the planning period

Dynamic Programming Scheme

- Define weights on E accordingly, that is, pay for the maintenance occasion at t' and the performed repairs at t
- Shortest path from $(0, T_1, \dots, T_n)$ to $(T+1, 0, \dots, 0)$ yields an optimal solution to (1)

Digraph, example



$n = 2,\ T = 8,\ T_1 = 3,\ T_2 = 5$

Complexity

- Dijkstras algorithm runs in $\mathcal{O}(|V|log|V| + |E|)$ time
- $|V| = T\Pi_{i=1}^{n} T_i + 2$, $|E| \le n|V|$
- Looks very bad, however the digraph $\mathcal{D}$ is not connected
- Example: if $T_i = T - 1$, and T is large, then $|V|$ is huge, but the connected component of $(0, T_1, \dots, T_n)$ is small. In fact $|V_0| = 3$
- Less dependent on discretization

A* algorithms

- Improves on Dijkstras algorithm by computing a heuristic estimate $h(x),\ x \in V$ of $d(x, \text{goal})$, distance to goal, thus guiding the shortest path search.
- In this case, an admissible heuristic for computing remaining distance can be based on the LP relaxation of (1)
- Three flavors, *a priori* lower bounds on LP solution, plain vanilla LP and strengthened LP formulations
- Relaxations strong for short horizons

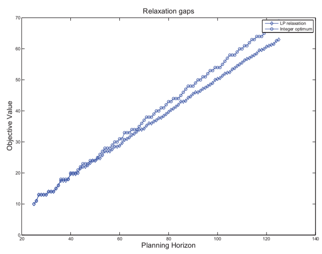
A* primer

- Maintains a priority queue for a subset of nodes called the open nodes
- Score is actual cost from source to node plus estimated costs of node to goal
- Expands the open node with the best score first
- Computes new scores of the neighbours of that node, and adds them to the open nodes.
- When the goal node is expanded, the search is complete, and path optimal if the estimated costs are admissible

Heuristics

- A^* formulation immediately gives strong heuristics for (1), based on limiting the open nodes/memory usage
- Heuristic is strong if the ordering of scores is similar to the ordering of optimal values
- Restrict the maximum number of open nodes at a given t
- Introduce a memory horizon, that is, if we have expanded a state far into the future, forget all nodes far into the past

Heuristics, motivation

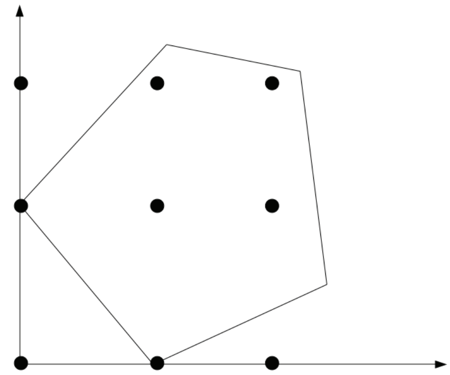


Relaxation gaps increase approximately linearly in the planning horizon

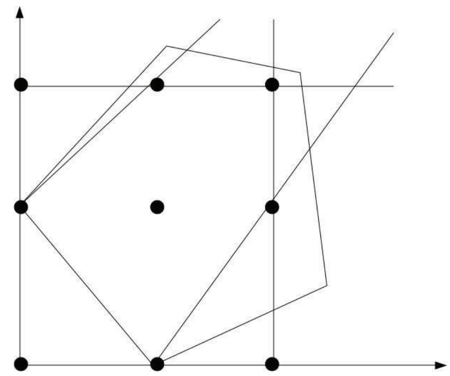
Strengthening Relaxations

- Classical mathematical programming methods to derive cutting planes of (1)
- Zero-half Chvatal-Gomory cuts, combinatorial cuts
- Polynomial-time partial separation algorithm

Cutting planes



Cutting planes



Zero-half cuts

Basic observation, if

$$\sum_j a_{ij} y_j \geq b_i \quad (4)$$

are valid for the replacement problem, then so is

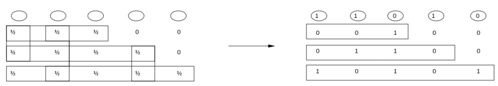
$$\sum_j \left\lceil \sum_i u_i a_{ij} \right\rceil y_j \geq \left\lceil \sum_i u_i b_i \right\rceil \quad (5)$$

for any $u_i \geq 0$. Called a zero-half cut if $u_i \in \{0, 1/2\}$

Zero-half cuts, meta-recipe

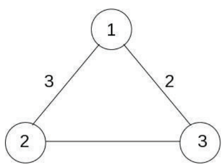
- Pick an odd number of constraints from (1b) with multiplier $u_i = 1/2$
- 'Mix' them together using (1c)
- From this, construct a graph: chosen constraint from (1b) is a node, and there is an edge if they were mixed, or if they share a common variable.
- Certain technical rules on how to mix
- An edge is said to cross a node k if the mixing overlaps in time with the constraint k.

Zero-half cuts



$$(z_1 + x_{31}) + (z_2 + x_{22}) + (x_{13} + x_{23} + x_{33}) + z_4 + z_5 \geq 2 \tag{6}$$

Zero-half cuts, example



Zero-half cuts, example

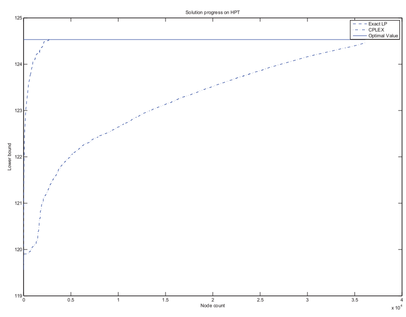
Proposition 1 *If for every k the subgraph that is obtained by deleting node k can be partitioned into subgraphs of size 2, such that every edge crossing k is in the partitioning, then the inequality is a facet of the replacement polytope, if some additional technical assumptions are made.*

This is very similar to the odd cycle inequalities for set packing/covering polytopes. We also note that for some graphs containing bad crossings, the corresponding inequality is still valid, and can be lifted to facets

Combinatorially obtained facets

- Basic concept is to envelop shortlived component into components with longer life.
- Partial overlap with the zero-half cuts
- Subclass can be generated as many many-to-many shortest path problems in polynomial time
- Do not seem to generalize to more complex models

Numerics



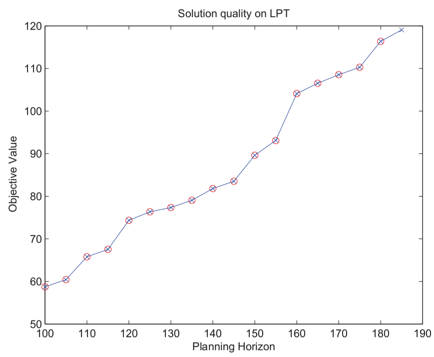
Solution progress for a High Pressure Turbine (VAC) in the number of explored nodes. $n = 9$, $T = 200$

Numerics

	CPLEX		Simple		LP		Strong LP	
	Time(s)	Nodes	Time(s)	Iterations	Time(s)	Iterations	Time(s)	Iterations
1	0.94	187	578	46457	484	13019	30	146
2	1.70	503	0.4	521	6.6	81	14	51
3	3600*	326947	1.7	3067	198	1883	2327	3543

The strenghtened LP uses CPLEX root node processing to generate zero-half cuts

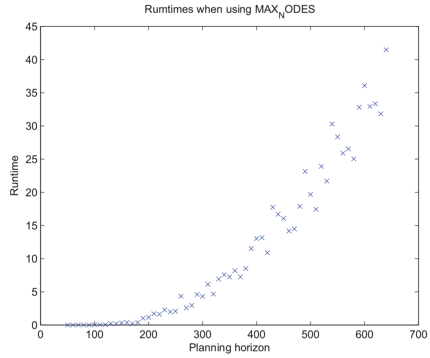
Heuristics



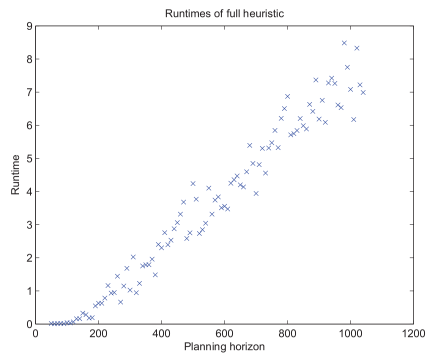
Heuristics

- The full heuristic solves all tested problems where optimum is known to less than 1% within optimality. In most cases the optimum is found
- Runtime is in practice linear in T for the full heuristic. If we choose to only limit the number of nodes at t, seems to be in practice quadratic

Heuristics



Heuristics



Extensions

- Why did the constructions work?
- Markovian property
- Constraints had 0/1 coefficients, and 0/1 right hand sides
- Many variants of the basic problem can be formulated in this way

Extensions, example, component individuals

- Instead of considering $i \in \mathcal{N}$ as component, we think of them as tasks that must be performed
- Possibly many decisions with different costs and lives that satisfy task i
- Canonical example is that given a task failure to repair or replace
- Modify parameters into $T_i^r,\ r \in R_i$ and variables into x_{it}^r, z_t

Extensions, example, component individuals

$$\min_{(x,z)} \sum_{i \in \mathcal{N}, t \in \mathcal{T}, r \in R_i} c_{it}^r x_{it}^r + \sum_{t \in \mathcal{T}} d_t z_t \tag{7a}$$

$$\text{s.t.} \sum_{r \in R_i} \sum_{t \in \max(l - T_i^r, 0)}^{l} x_{it}^r \geq 1 \qquad i \in \mathcal{N}, l \in \{\tau_i + 1, \ldots, T\} \tag{7b}$$

$$\sum_{r \in R_i} \sum_{t=0}^{\tau_i} x_{it}^r \geq 1 \qquad i \in \mathcal{N} \tag{7c}$$

$$x_{it}^r \leq z_t \qquad i \in \mathcal{N}, t \in \{1, \ldots, T\}, r \in R_i \tag{7d}$$

$$x_{it}^r, z_t \in \{0, 1\} \qquad i \in \mathcal{N}, t \in \mathcal{T}, r \in R_i \tag{7e}$$

Conclusions

- Dynamic programming approaches can be superior to general MIP solvers
- Insensitive to time discretization
- Fairly general zero-half cut procedure will generate cutting planes in a manner similar to odd cycle inequalities for set covering problems

Thank you!