Optimal Crane Scheduling

Ionuț Aron
IBM Watson Lab
Latife Genç Kaya, John Hooker
Carnegie Mellon University
Iiro Harjunkoski, Marco Fahl
ABB Group

November 2006

PITA

Pennsylvania Infrastructure Technology Alliance
A Commonwealth, University & Industry Partnership

CONTACT US!

Co-Directors
Gary Fedder, Richard Sause

Associate Directors
Matthew Sanfilippo
Robert Alpago

Carnegie Mellon
Institute for Complex Engineered Systems
LEHIGH University
ATLSS
DCED Pennsylvania

Thanks to…

- PITA – Pennsylvania Infrastructure Technology Alliance.
- ABB Group

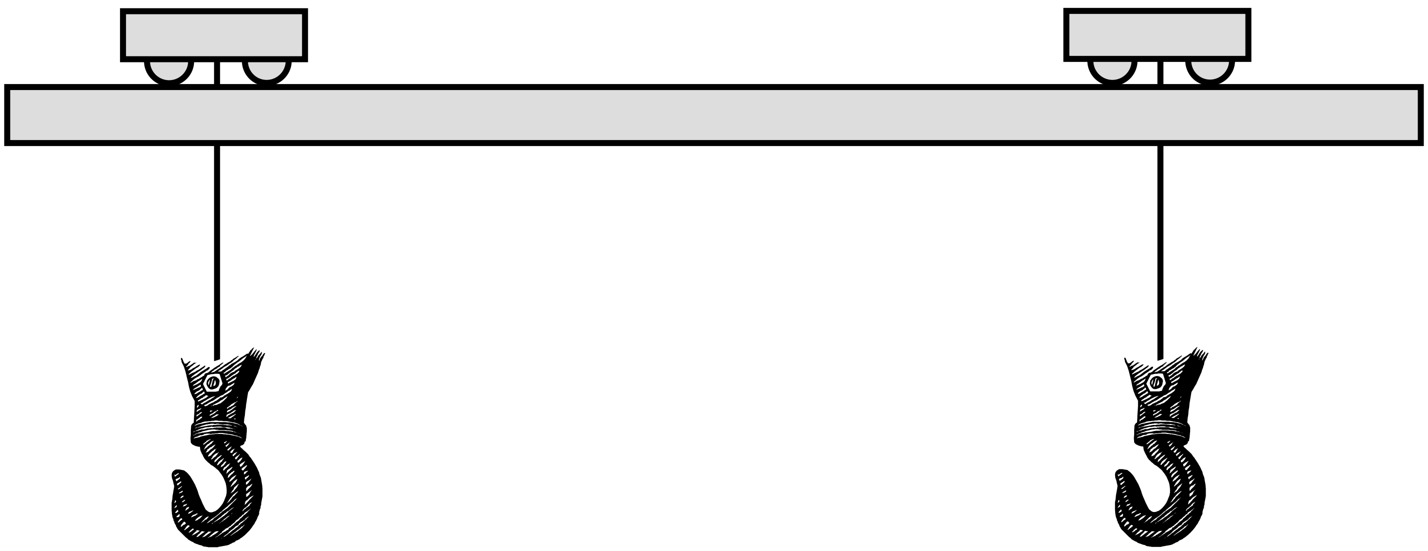
Problem

- Schedule 2 cranes to transfer material between locations in a manufacturing plant, e.g. copper processing.

PTIA

Problem

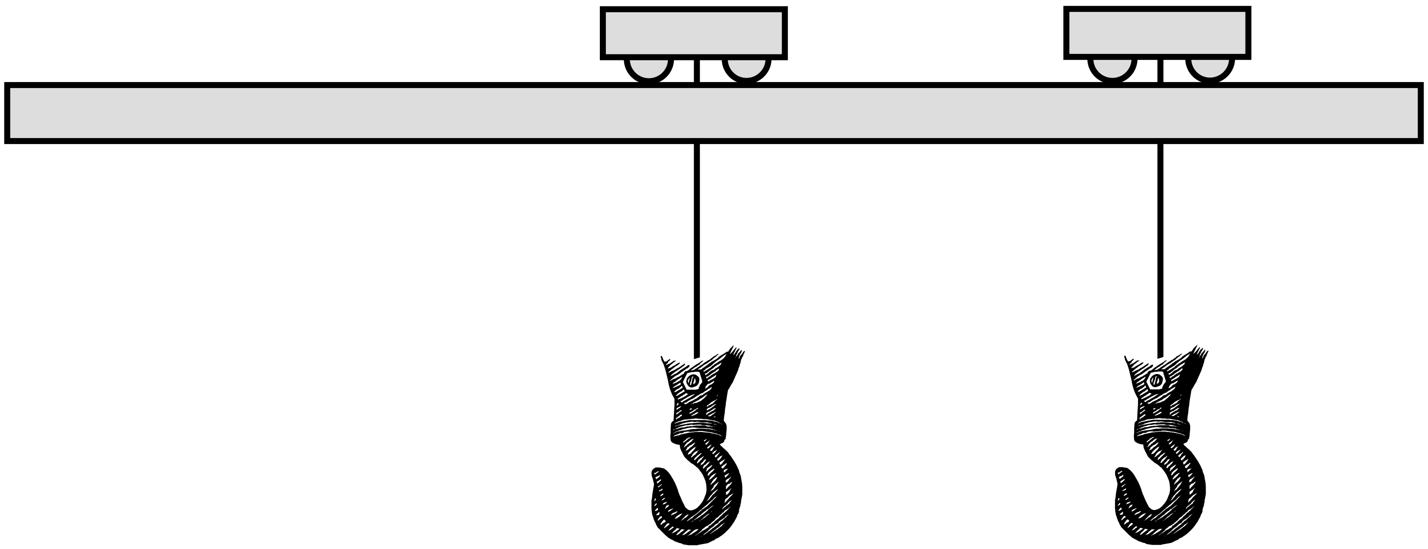
– Cranes can move on a common track.



PTA

Problem

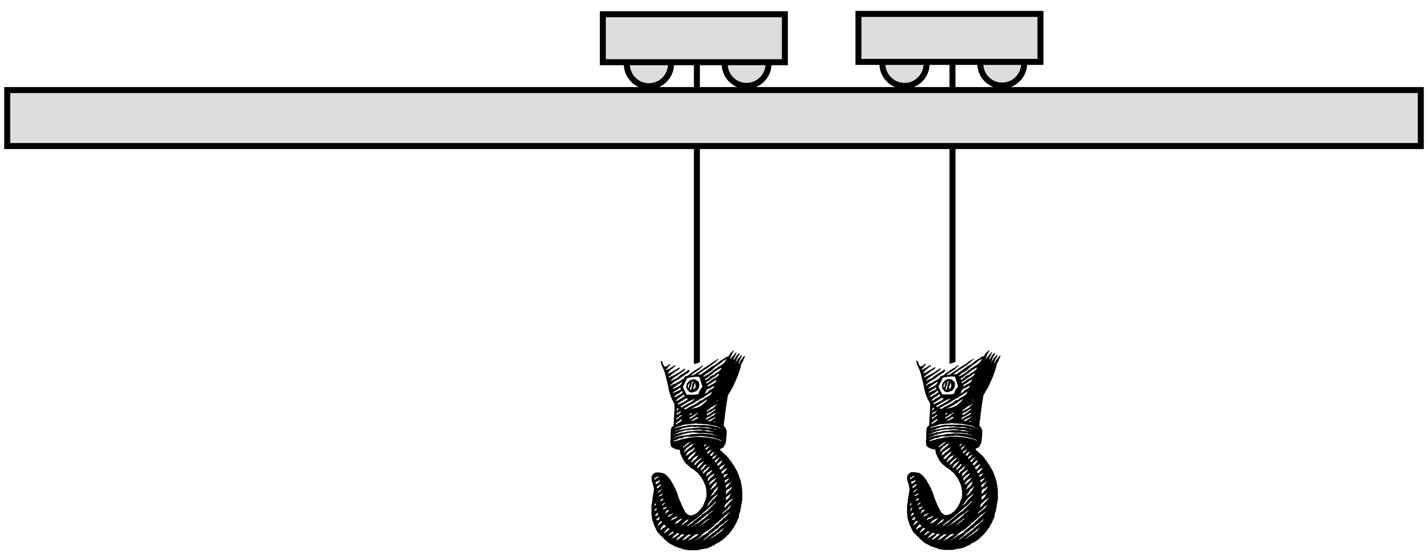
– Cranes cannot pass each other.



PTTA

Problem

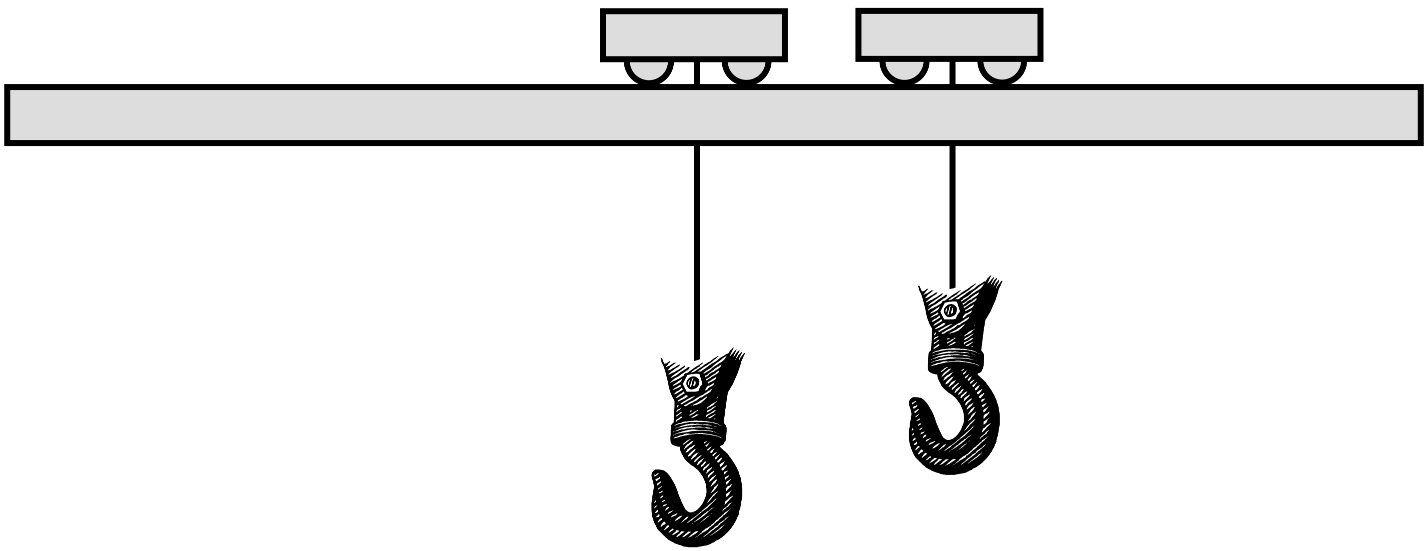
– Vertical movement.



PIIA

Problem

– Vertical movement.
  - Horizontal and vertical movement can be simultaneous.



Problem

- Constraints:
  - As many as 300 jobs, each with a time window and priority.
  - Precedence relations between jobs.
  - A job may require several stops.
- Objective: minimize total penalty
  - Penalties reflect deviation from desired start times or completion times.
  - Main objective is to follow production schedule as closely as possible.

PIIA

Problem

- Three problems in one:
  – Assign jobs to cranes.
  – Find ordering of jobs on each crane.
  – Find space-time trajectory of each crane.
    - Crane scheduling problems are coupled since the cranes must not pass one another.

Two-phase Algorithm

- Phase 1: Local search
  - Assign jobs to cranes
  - Sequence jobs on each crane
  - Solve simultaneously by tabu-like local search.
- Phase 2: Dynamic programming (DP)
  - Find optimal space-time trajectory for the cranes.
  - Solve for the two cranes simultaneously.
  - One crane can yield to another.

Local Search

- Neighborhood is defined by two types of moves.
  - Change assignment
  - Change sequence
- Evaluation of moves
  - Use an approximate evaluation function to limit neighborhood.
  - Check best moves with DP (phase 2).

Dynamic Programming

- Find optimal space-time trajectory for each crane.
  - Sequence of jobs on each crane is given.
  - Minimize sum of penalties, which depend on pickup and delivery times.

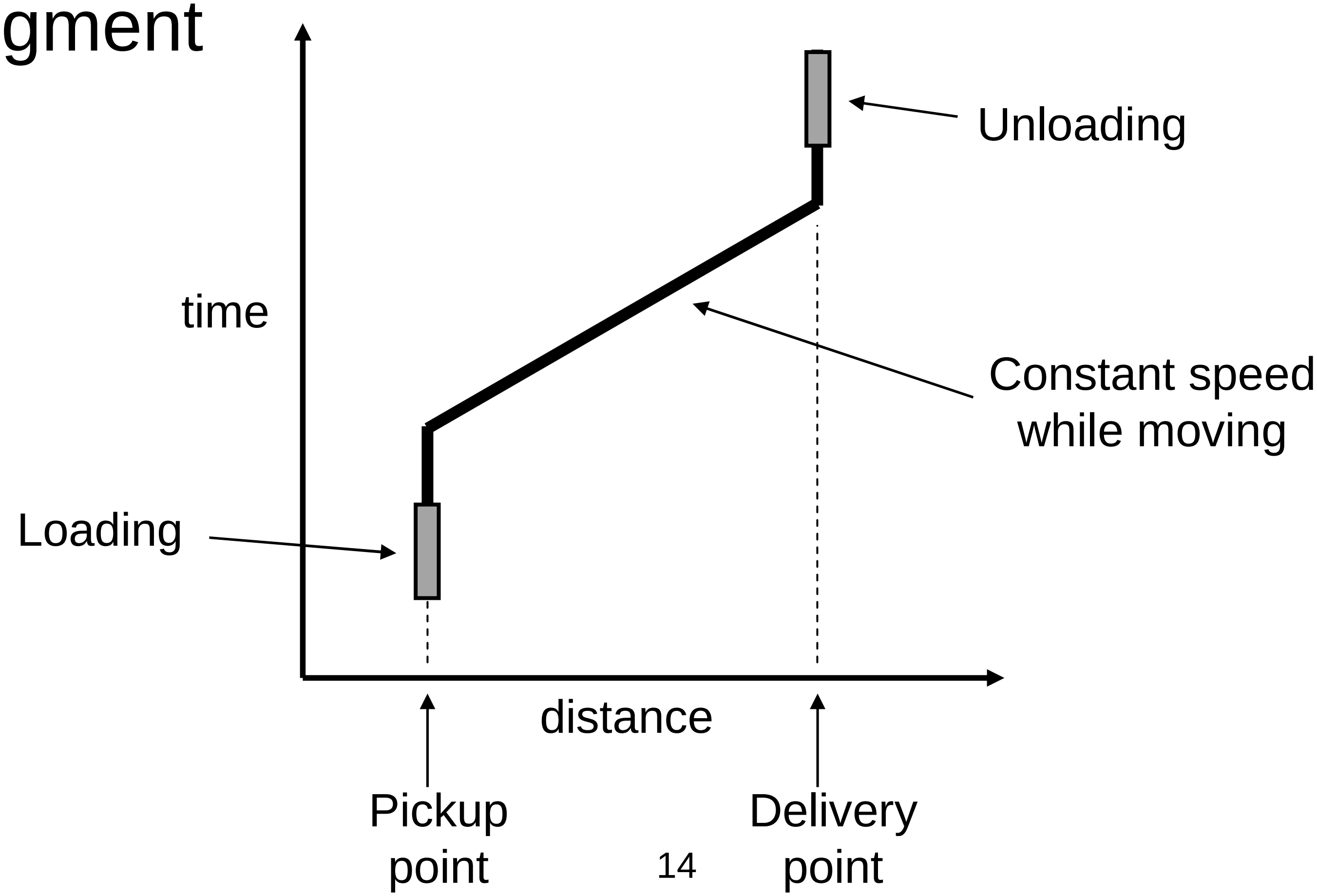
Dynamic Programming

- Each job consists of one or more "segments."
  - Order of segments within a job is fixed.
- Each segment consists of loading, movement to another position, unloading.
- Given for each segment:
  - Loading and unloading positions.
  - Time required to load, unload.
  - Min time for crane movement.

PITA

Dynamic Programming

- Space-time trajectory of a crane for one segment



PTA

Dynamic Programming

- High resolution needed to track crane motion.
  - Time horizon: hours
  - Granularity: 10-second intervals
    - Thousands of discrete times
  - Cranes are stationary most of the time
    - While loading/unloading
    - But motion is fast when it occurs (e.g. 1 meter/sec)
  - Feasibility is main issue
    - No obvious role for approximate DP

Dynamic Programming

- Main issue: size of state space.
- State variables:
  - Position of each crane on track.
  - How long each crane has been loading/unloading.
  - Current segment in process for each crane
    - Negative number if on the way to load the segment.
    - Positive number if loading, unloading, or on the way to unload.

Dynamic Programming

- DP recursion:

$$f_{t+1}\begin{pmatrix} x_{\cdot,t+1} \\ u_{\cdot,t+1} \\ s_{\cdot,t+1} \end{pmatrix} = \min_{\begin{pmatrix} x_{\cdot t} \\ u_{\cdot t} \\ s_{\cdot t} \end{pmatrix} \in S \begin{pmatrix} x_{\cdot,t+1} \\ u_{\cdot,t+1} \\ s_{\cdot,t+1} \end{pmatrix}} \left\{ f_t\begin{pmatrix} x_{\cdot t} \\ u_{\cdot t} \\ s_{\cdot t} \end{pmatrix} + \sum_c g_{ct}\left(s_{ct}, s_{c,t+1}\right) \right\}$$

Position

Loading/
unloading
time

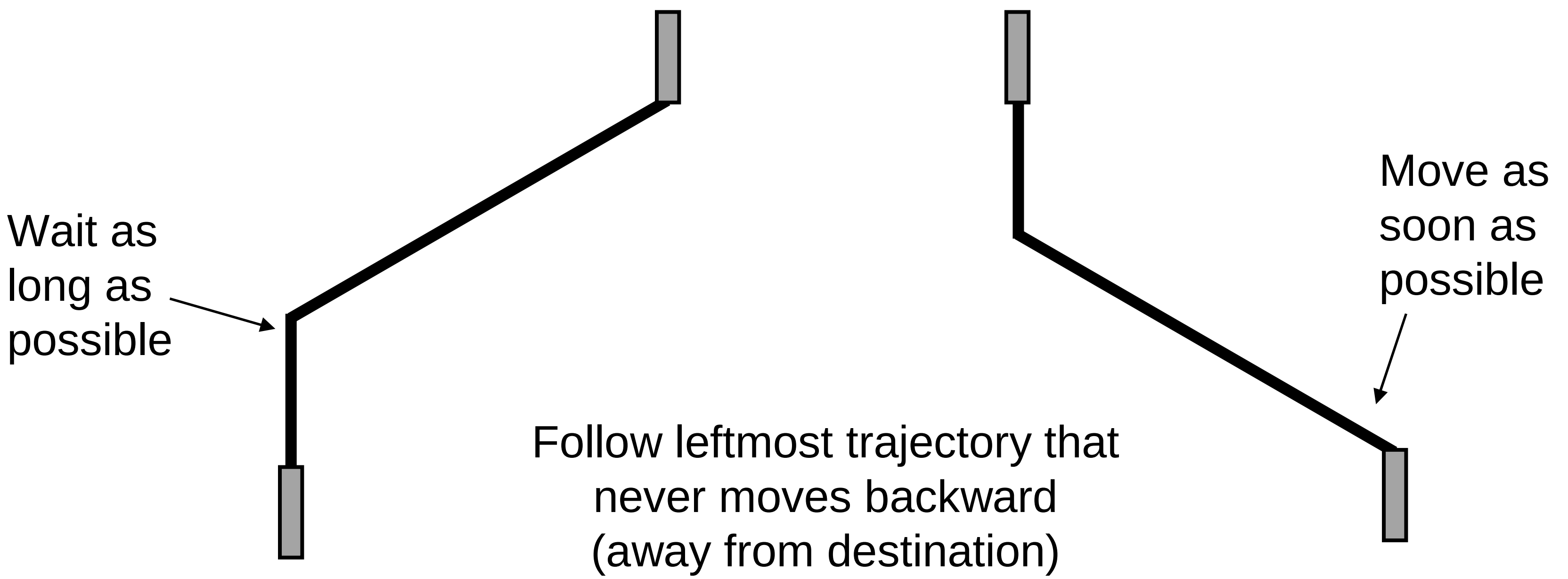
Segment

Transitions that satisfy constraints

Computes
penalty

Dynamic Programming

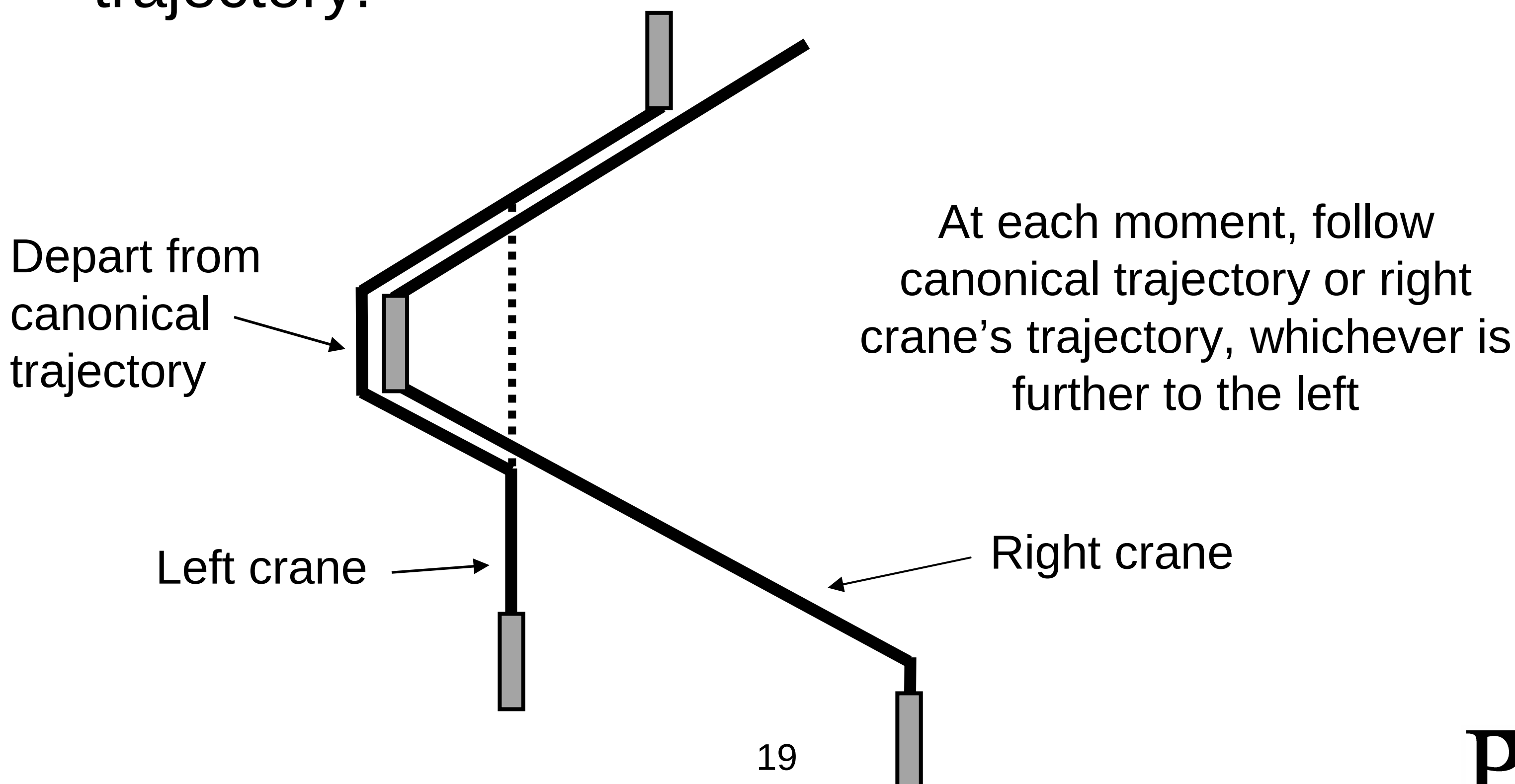
- State space reduction
  - *Canonical* trajectory for the left crane:



PITA

Dynamic Programming

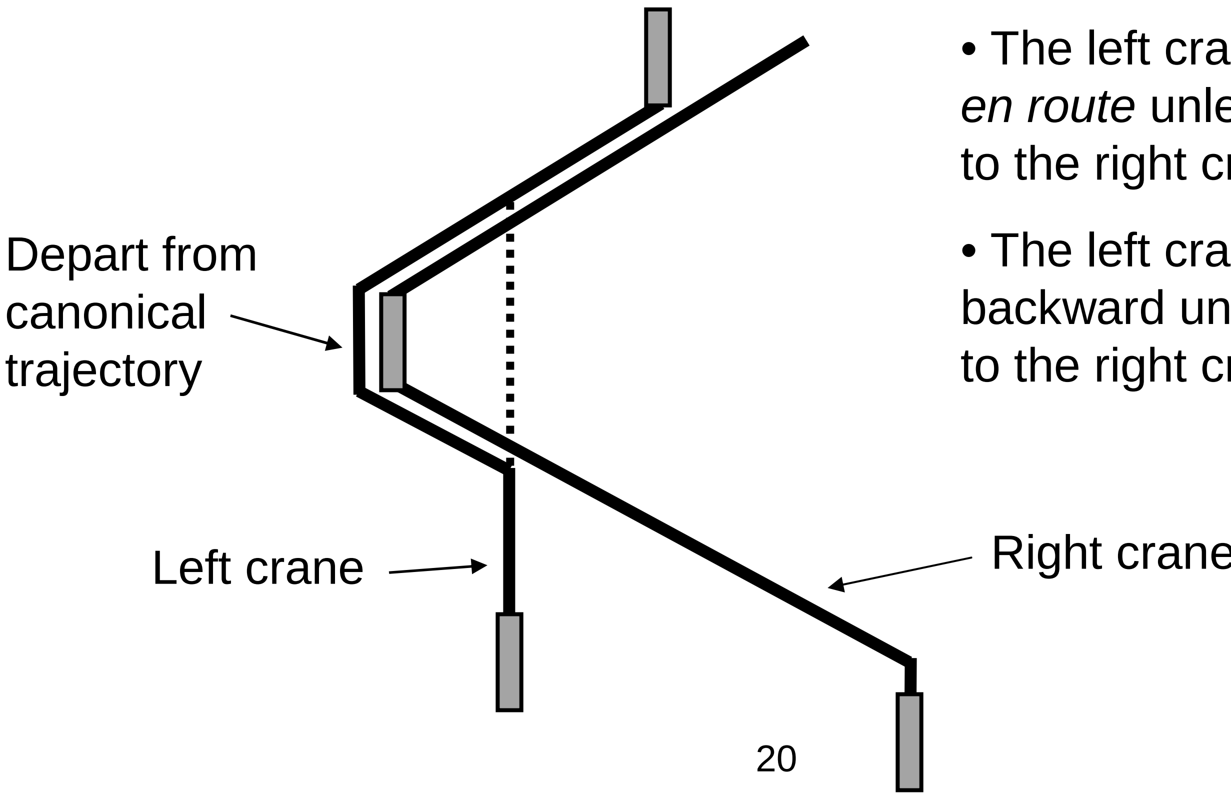
- State space reduction
  - *Minimal* trajectory for left crane, given right crane trajectory:



PITA

Dynamic Programming

- State space reduction
  - Properties of the minimal trajectory:



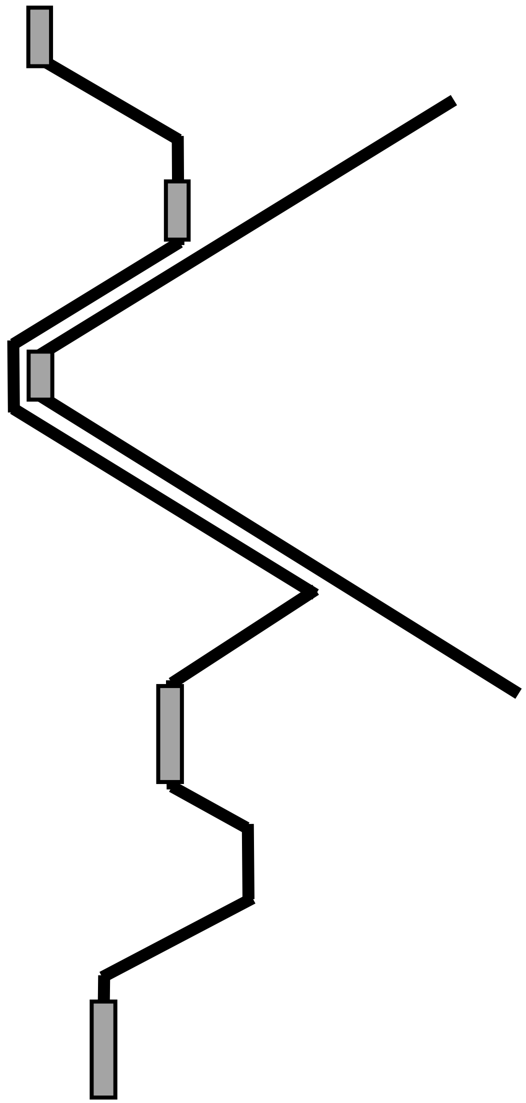
• The left crane never stops *en route* unless it is adjacent to the right crane.

• The left crane never moves backward unless it is adjacent to the right crane.

PTA

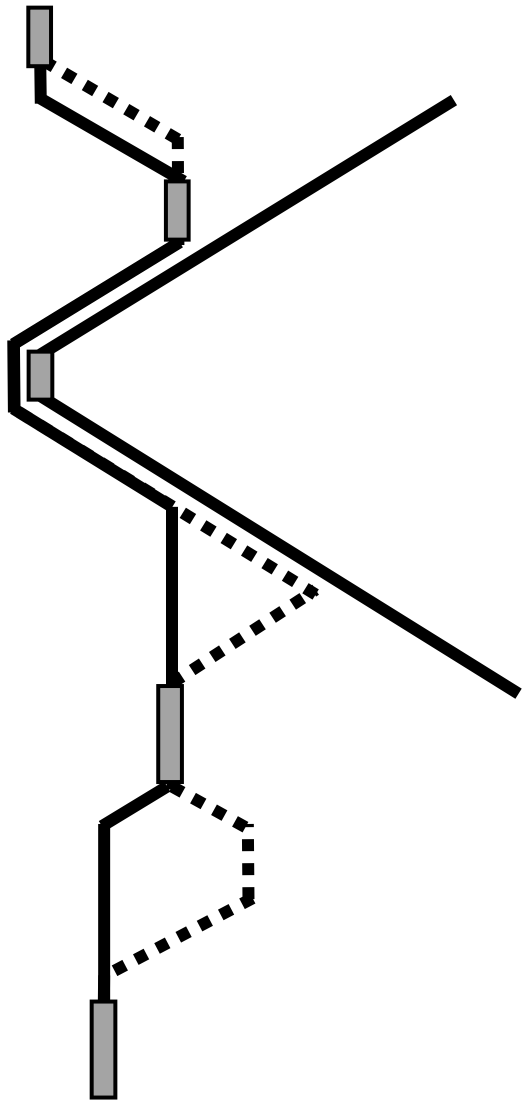
Dynamic Programming

- State space reduction
  - **Theorem:** Given optimal trajectories for both cranes, either crane's trajectory in each segment can be replaced by a minimal trajectory without sacrificing optimality or feasibility.



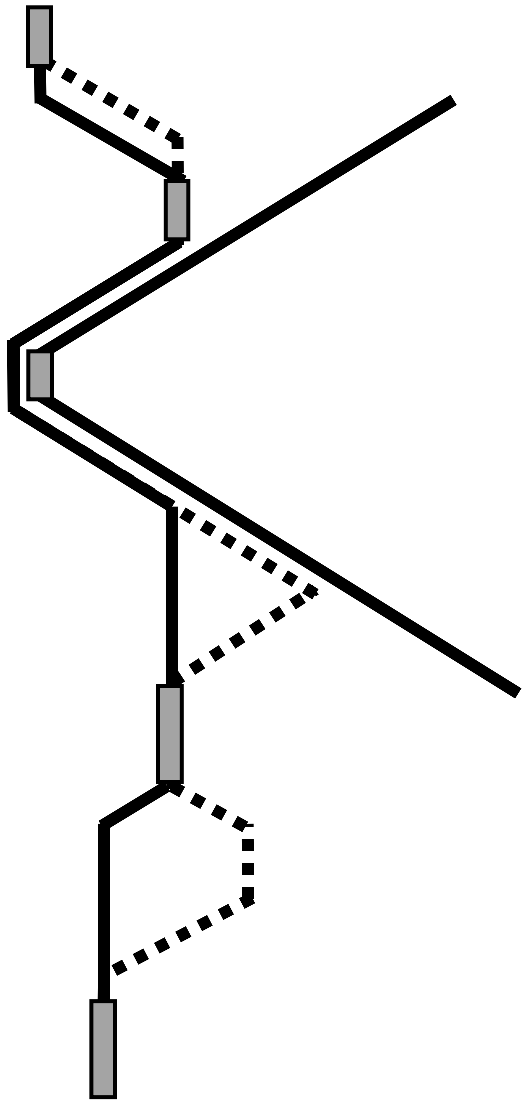
Suppose these segments are part of an optimal trajectory.

PTTA



Suppose these segments are part of an optimal trajectory.

Change left crane to its minimal trajectory.

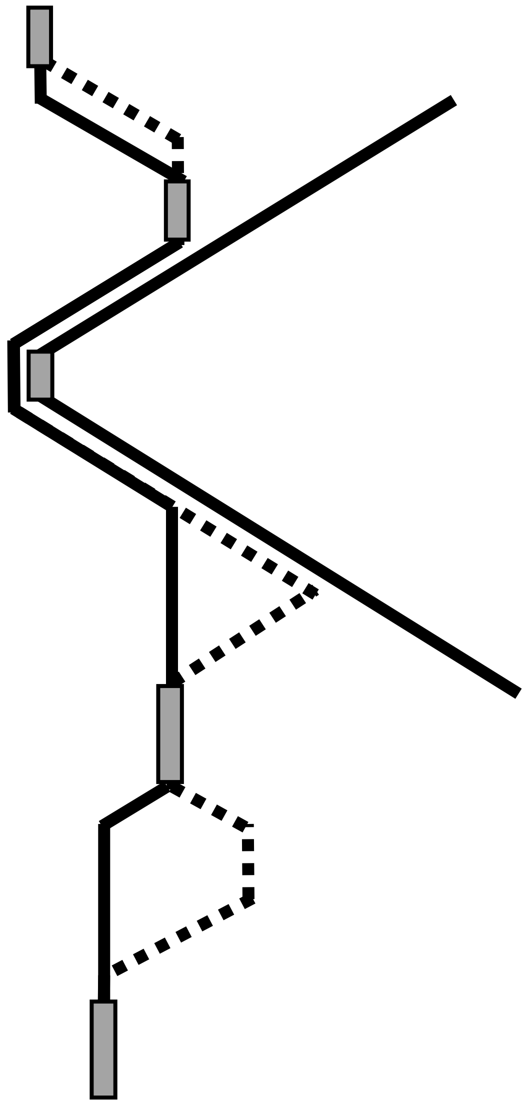


Suppose these segments are part of an optimal trajectory.

Change left crane to its minimal trajectory.

This is still feasible, because

- there is no interference from right crane
- pickup and delivery times are unchanged.



Suppose these segments are part of an optimal trajectory.

Change left crane to its minimal trajectory.

This is still feasible, because

- there is no interference from right crane
- pickup and delivery times are unchanged.

Cost is the same if it depends only on pickup & delivery times.

PITA

Dynamic Programming

- State space reduction
  - **Corollary:** Exclude state transitions in which a crane stops en route, or moves backward, unless it is adjacent to the other crane.

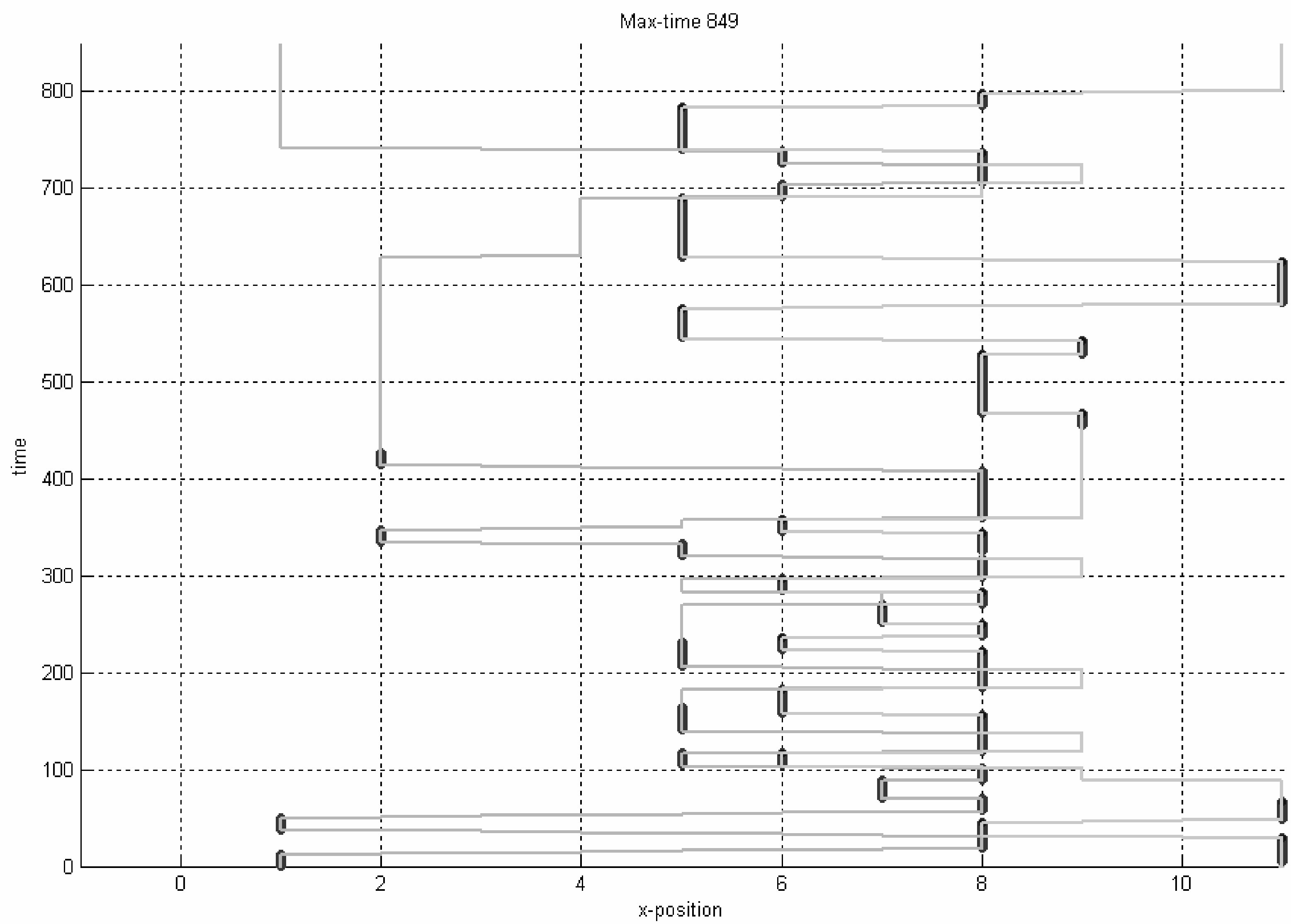
PITA

Dynamic Programming

- State space reduction
  - Since the main objective is to follow the production schedule, fairly tight time windows can be used.
  - This reduces the number of jobs in the state space at any one time.
  - If necessary, time horizon can be split into segments to be solved separately.

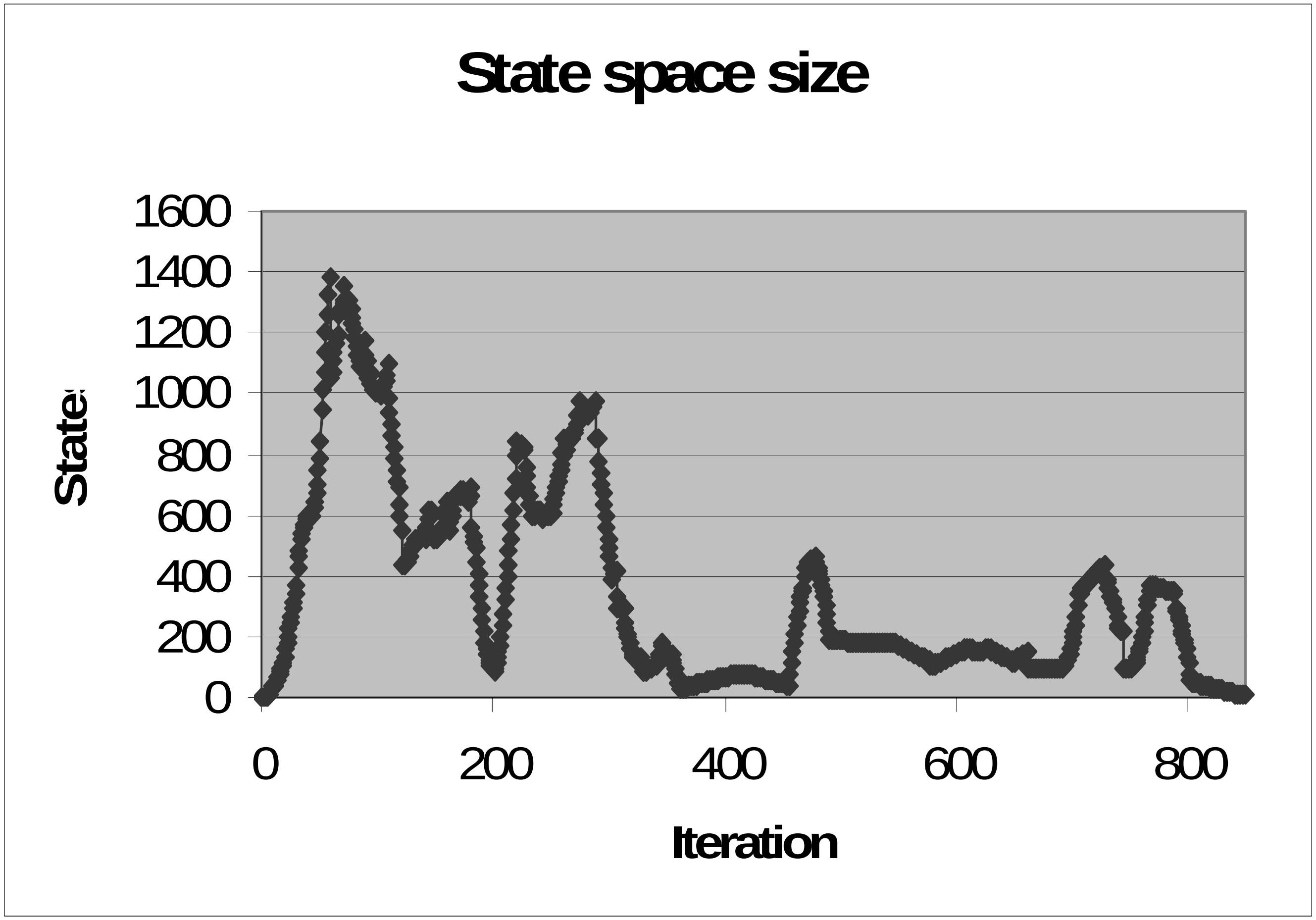
Dynamic Programming

- Preliminary computational results.
  - 10 crane positions (realistic).
  - Penalize time lapse between release time and pickup.
  - Minimize sum of penalties.
  - Theoretical maximum number of states in any given stage is approximately $10^8 – 10^9$.



15 jobs, 30 segments

PITA


State space size
States
1600
1400
1200
1000
800
600
400
200
0
0
200
400
600
800
Iteration

PTA

Precedence Constraints

- The crane problem has more complex precedence constraints than ordinarily occur in scheduling problems.
  - **Hard** precedence constraints apply to **groups** of jobs.
  - **Soft** precedence constraints are enforced by imposing penalties.
  - Some precedence constraints are enforced by **combinations** of hard and soft constraints.

Precedence Constraints

- Hard constraint
  - Let S, T be sets of jobs.
  - $S < T$ means that all jobs in S assigned to a given crane must run before any job in T assigned to that crane.
    - Jobs within S may occur in any order, similarly for T.
    - {cranes that perform the jobs in S }
= {cranes that perform the jobs in T }
    - Jobs in neither S nor T can run between S and T.

PITA

Precedence Constraints

- Soft constraint
  - Assign penalty to gap between release time and start time of a job.
  - If release time of job i precedes release time of job j, we can impose a soft $i < j$ by assigning high penalties to both jobs.

PITA

Precedence Constraints

- Consecutive jobs
  - Suppose $S < T$ and S, T should occur consecutively.
    - There should be no jobs i, j, k assigned to the same crane such that $i \in S$, $j \notin S \cup T$, $k \in T$, and j runs between i and k.
  - To enforce this:
    - Impose $S < T$.
    - Give jobs in S very similar release times to jobs in T.
    - Impose high penalties on jobs in S, T.

Assignment and Sequencing

- Heuristic algorithm:
  - Form initial assignment/sequencing with simple nearest job heuristic.
  - Call DP repeatedly with increasing time windows until feasible solution is found or max time windows reached.
  - Move to random neighboring solution
    - Consider neighbors with *estimated* penalty within 15-20% of best solution found so far.

Assignment and Sequencing

- Penalty estimate:
  - Assume each job finishes at EFT + 10%.
- Local search
  - Select randomly from the following:
    - Move a random job from one crane to the other
    - Swap cranes for a random pair of jobs
    - Swap positions of a random pair of jobs on one crane

Combined Algorithm

Preliminary Computational Results

Jobs	Iterations to first feasible solution	Time to feasible solution* (sec)
12	2	1
24	3	3
28	3	3

*Only one feasible solution found.

PITA

Future Work

- Generate nogood constraints from DP solution.
  - Identify a subsequence of jobs (assigned to the same crane) that is responsible for infeasibility.
  - Create a nogood constraint that prevents heuristic phase from assigning this subsequence to the same crane again.

PIIA