Restrict and Relax Search

George Nemhauser
Georgia Tech

Menal Guzelsoy, SAS
Martin Savelsbergh, Newcastle

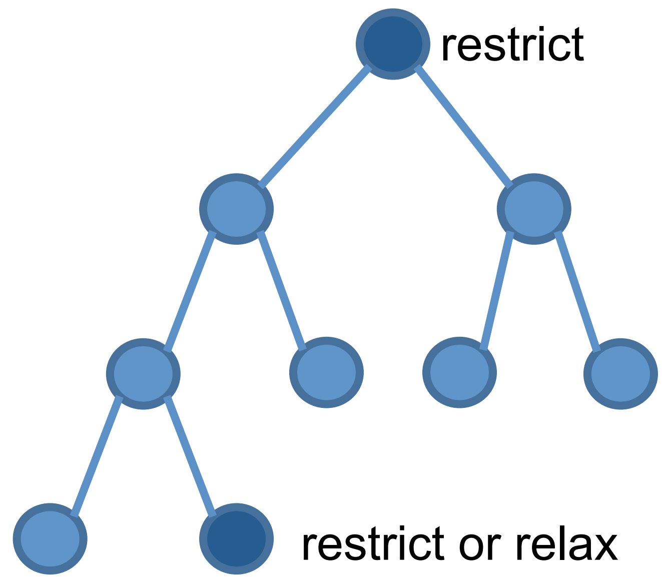
MIP 2013, Madison, July 2013

Outline

- Motivation
- Restrict-and-Relax Search
  - Initial restriction
  - Fixing and unfixing
- Computational experiments
  - 0-1 integer programs
  - Multi-commodity fixed charge network flow
  - Maritime inventory routing

Restrict-and-Relax Search

- What is it?
  - Branch-and-bound algorithm that **always** works on a restricted integer program
  - Branch-and-bound algorithm that uses **local information** to decide whether to relax (unfix variables) or restrict (fix variables)



Why do it?

- **Restrict** is for improved efficiency

  (Solve smaller MIPs as in RINS or in MIP based neighborhood search algorithms for specific problems)

- **Relax** is for improved quality

  (Like in column generation where new variables are added to the problem)

Goal

- Get good feasible solutions to very large MIPs quickly.
- Retain the possibility of getting a provably good bound or optimality.

Restrict-and-Relax Search

Original IP

$$
\begin{aligned}
z = \ & \min cx \\
& Ax = b \\
& x \in \mathbb{B}^r \times \mathbb{R}^{n-r}
\end{aligned}
$$

Restricted IP

$$
\begin{aligned}
z_F = \ & \min cx \\
& Ax = b \\
& x_i = \bar{x}_i \ , \ i \in F \\
& x \in \mathbb{B}^r \times \mathbb{R}^{n-r}
\end{aligned}
$$

Restricted IP at node of the search tree

$$
\begin{aligned}
v_t = \ & \min cx \\
& Ax = b \\
& x_i = \bar{x}_i \ , \ i \in F \cup B_t \\
& x \in \mathbb{B}^r \times \mathbb{R}^{n-r}
\end{aligned}
$$

Restrict-and-Relax Search

Restricted IP at node of the search tree

$$\begin{aligned} v_t = \ & \min cx \\ & Ax = b \\ & x_i = \bar{x}_i \ , \ i \in F \cup B_t \\ & x \in \mathbb{B}^r \times \mathbb{R}^{n-r} \end{aligned}$$

Modified restricted IP at node of the search tree

$$\begin{aligned} \bar{v}_t = \ & \min cx \\ & Ax = b \\ & x_i = \bar{x}_i \ , \ i \in \bar{F} \cup B_t \\ & x \in \mathbb{B}^r \times \mathbb{R}^{n-r} \end{aligned}$$

Goal: Choose $\bar{F}$ in such a way that $\bar{v}_t < v_t$

Restrict-and-Relax Search

- Key decisions
  - How to define the initial restricted integer program?
  - How to determine the variables to fix or unfix?
  - At which nodes in the search tree to relax or restrict?

How to define the initial restricted integer program?

- Based on a known feasible solution
- Based on the solution to the LP relaxation
- Based on the Phase I solution to the LP relaxation

How to define the initial restricted integer program?

Scheme: variables binary in LP solution are fixed

- $x_{LP} = 0$ for variable: Score: c
- $x_{LP} = 1$ for variable: Score: -c
- Fix variables from large to small scores (Fix variables that if fixed to opposite value would increase the objective function the most.)
- Fix at most 90% of variables.

How to determine the variables to fix and unfix?

- Fixing variables (LP feasible node):

$$\boxed{\text{if } x_i^* = 0, \text{ then } r_i^* \geq 0 \text{ and if } x_i^* = 1, \text{ then } r_i^* \leq 0}$$

Choose variables to fix in nondecreasing order of absolute value of reduced costs

Fix variables in the current primal solution that are unlikely to change value in an optimal LP solution.

- Unfixing variables (LP feasible node):

$$\boxed{\text{if } x_i^* = 0 \text{ and } r_i^* < 0 \text{ or if } x_i^* = 1 \text{ and } r_i^* > 0}$$

Choose variables to unfix in nondecreasing order of absolute value of reduced costs

Unfix variables in the current primal solution that are likely to result in an optimal solution with lower LP value.

How to determine the variables to fix and unfix?

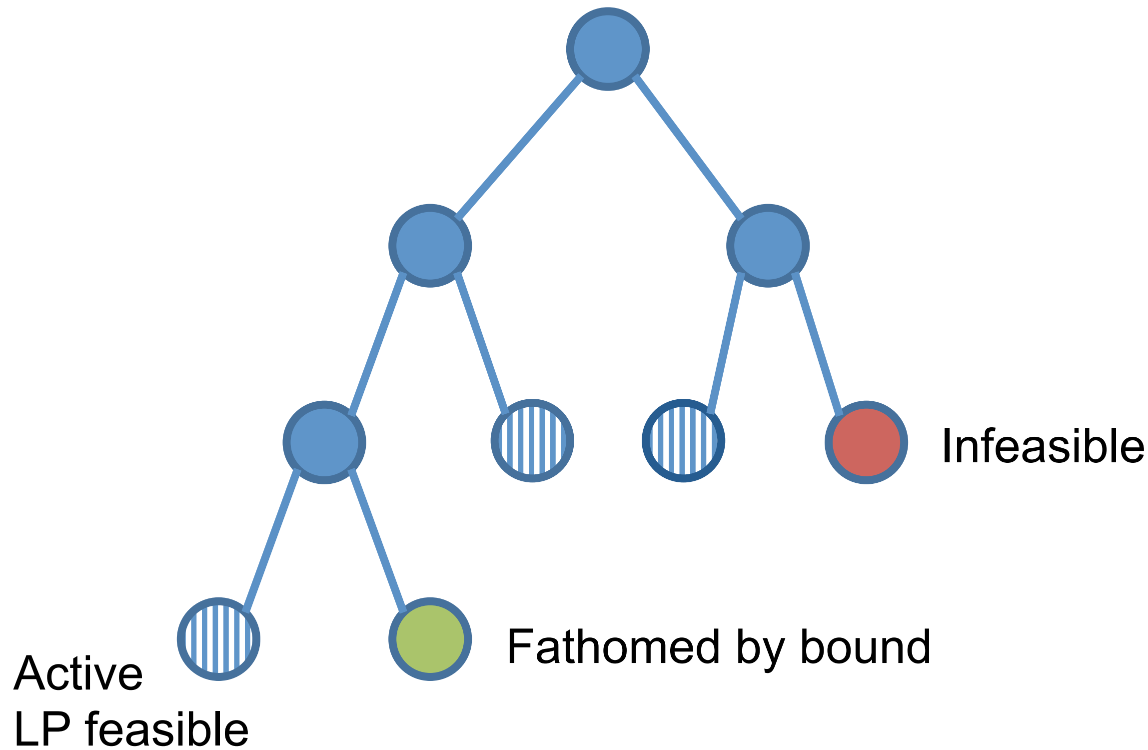
- Implementation - Gradual transition:

$$\min cx$$
$$Ax = b$$
$$x_i = \bar{x}_i \ , \ i \in F_t^j \cup B_t$$
$$x \in \mathbb{B}^r \times \mathbb{R}^{n-r}$$

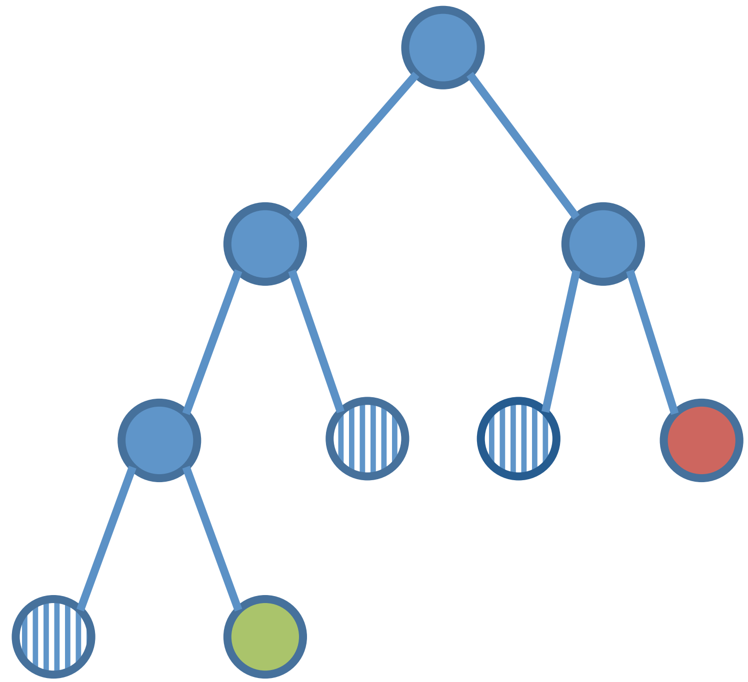
$$F_t^0 = F \text{ and } |F_t^j \setminus F_t^{j-1}| \text{ small}$$

  - Fast linear programming solves
  - Up to date dual information

Unfixing variables

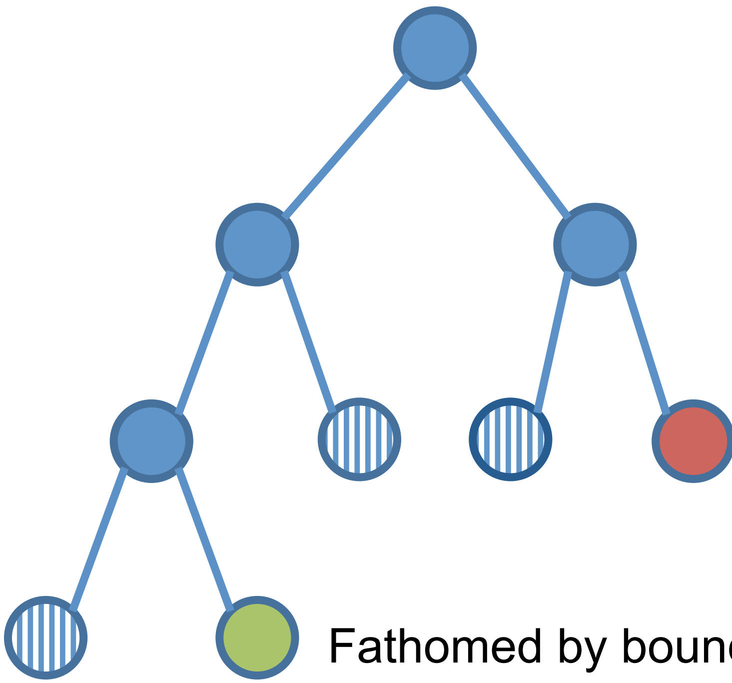


Unfixing variables



Opportunistic relaxing:
Unfix previously fixed variables

Unfixing variables



Infeasible:
(1) Resolve LP with all variables unfixed
(2) Unfix variables that change values

Fathomed by bound:
(1) Remove cut-off
(2) Resolve LP with all variables unfixed
(3) If value < best known, unfix based on reduced costs

Resolve LP with all variables unfixed guarantees that no nodes are discarded that could contain an optimal solution

At which nodes in the search tree to relax or restrict?

- Parameters:
  - level-frequency (l-f) : Relax/restrict at node t if node level is a multiple of l-f
  - unfix-ratio (u-r) : At each trial, unfix at most u-r % of the fixed variables
  - fix-ratio (f-r) : At each trial, fix at most f-r % of the free variables
  - node-trial-limit (t-l) : At each node, fix/relax at most t-l times
  - max-depth (max-d) : Fix/unfix only at nodes above tree level max-d
  - min-depth (min-d) : Fix/unfix only at nodes below tree level min-d
  - Pruned-by-bound (p-b) : If enabled, fix/unfix at nodes pruned by bound regardless of node level
  - Pruned-by-infeasibility (p-i) : If enabled, fix/unfix at nodes pruned by infeasibility regardless of node level

At which nodes in the search tree to relax or restrict?

- Default values:
  - level-frequency (l-f) : 4
  - unfix-ratio (u-r) : 5%
  - fix-ratio (f-r) : 2.5%
  - node-trial-limit (t-l) : 5
  - max-depth (max-d) : ∞
  - min-depth (min-d) : 0
  - Pruned-by-bound (p-b) : enabled
  - Pruned-by-infeasibility (p-i) : enabled

Computational Study

- Instances:
  - 127 selected 0-1 MIPs from MIPLIB 2010
  - Big restriction is eliminating 65 0-1 MIPs with fewer than 60% of the binary variables at 0-1 in LP optimal solution
  - Other eliminated: easy, infeasible, …
- Restrict-and-Relax
  - Initial restricted IP: based on LP relaxation
  - Default settings for parameters
  - Time limit: 500 seconds
- Implementation: SYMPHONY + CLP

Computational Study

- Default solver: Original IP
- Default solver: Restricted IP
- Restrict-and-relax Search

Results

	Original IP	Restricted IP
RR <	49	52
RR =	14	3
RR >	21	15
RR feas	10	24
RR no feas	2	0

By varying control parameters we can obtained improved solutions for all instances!

Results (sample)

	Original IP	Restricted IP	Restrict-and-Relax Search	Optimal
neos-693347	360	-	241	234
neos808444	-	-	0	0
m100n500k4r1	-22	-22	-24	-25

Results (sample)

	%fixed	#nodes	#unfix	avg.	#fix	avg.	#solutions
neos-693347	0.74	593	266	18	243	27	1
neos808444	0.9	633	129	24	1	1	1
m100n500k4r1	0.7	38075	11095	6	1620	12	4

Multi-Commodity Fixed-Charge Network Flow

Multi-commodity Fixed Charge Network Flow

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij}(d^k x_{ij}^k) + \sum_{(i,j) \in A} f_{ij} y_{ij}$$

Variable flow cost (>= 0)

Fixed cost of installing arc (>= 0)

Commodity flow balance

$$\sum_{j:(i,j) \in A} x_{ij}^k - \sum_{j:(j,i) \in A} x_{j,i}^k = \delta_i^k \quad \forall i \in N, \ \forall k \in K,$$

Arc capacity and coupling

$$\sum_{k \in K} d^k x_{ij}^k \leq u_{ij} y_{ij} \quad \forall (i,j) \in A,$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in A.$$

Do we install arc (i,j)?

$$x_{ij}^k \in \{0,1\} \quad \forall k \in K, \ \forall (i,j) \in A.$$

Does commodity k flow on arc (i,j)?

Computational Study

- Instances
  - Notation: T - #nodes(100x) - #arcs(1000x) - #commodities
  - Smallest (T-5-3-50)
    - 150,000 variables, 180,000 constraints, 750,000 non-zeroes
  - Largest (T-5-3-200)
    - 600,000 variables, 700,000 constraints, 3,000,000 no-zeroes

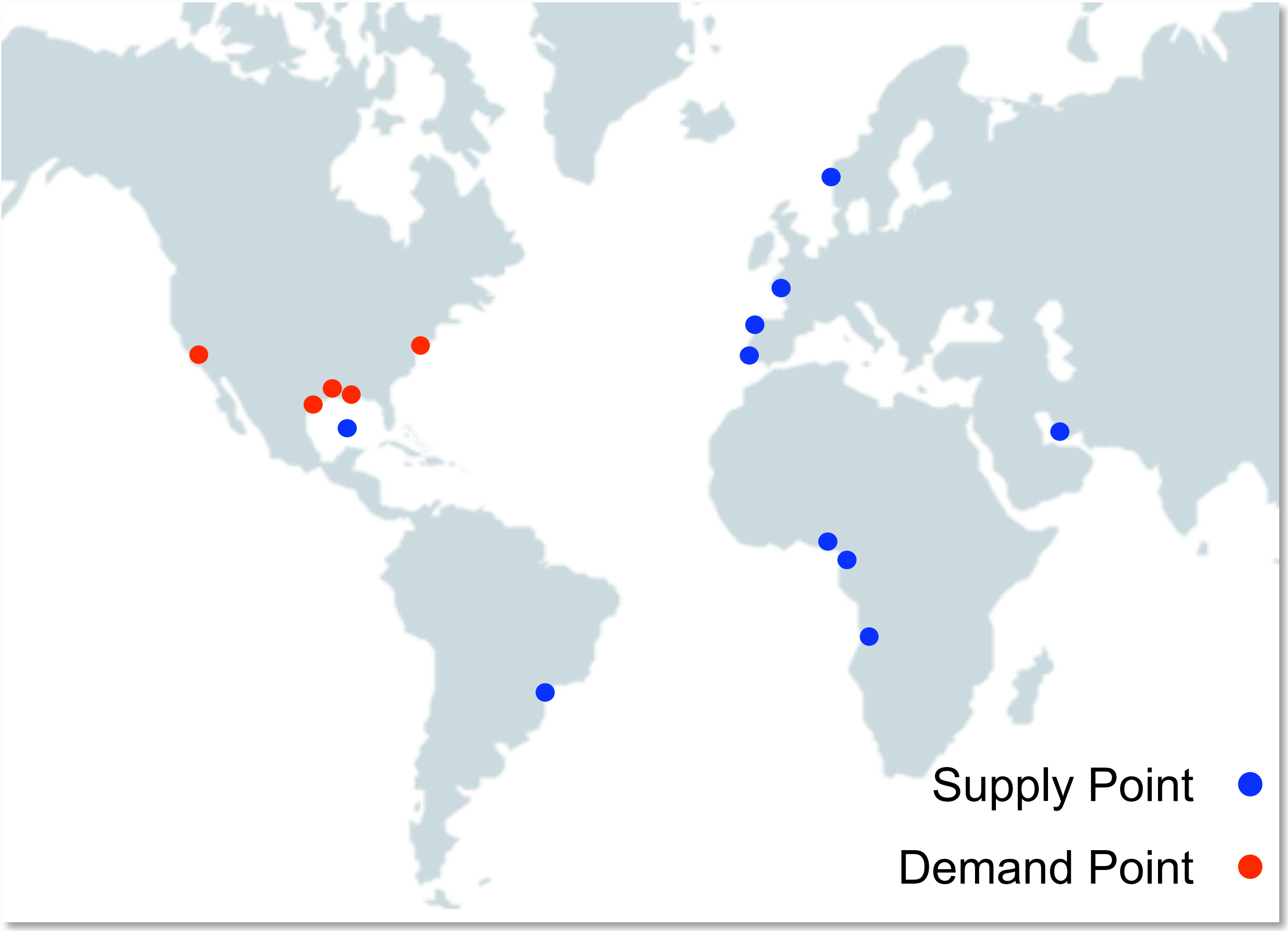
- Restrict-and-relax settings
  - Initial restriction:
    - **Phase I of simplex algorithm**, fix up to 90% of variables
  - Parameters:
    - unfix ratio: 6%, fix ratio: 5%
  - Time limit: 1 hour

Results

- With the original IP the LP was solved in only 5/11 instances
- With the original IP an integer solution was found for only 3/11 instances
- In 2 of those 3, the integer solution found was slightly better than the solution found by RR
- With the restricted IP feasible solutions were found but never a better solution than produced by RR and always with objective values more than twice that of RR
- RR produced on average around 100 solutions for each of the 11 instances

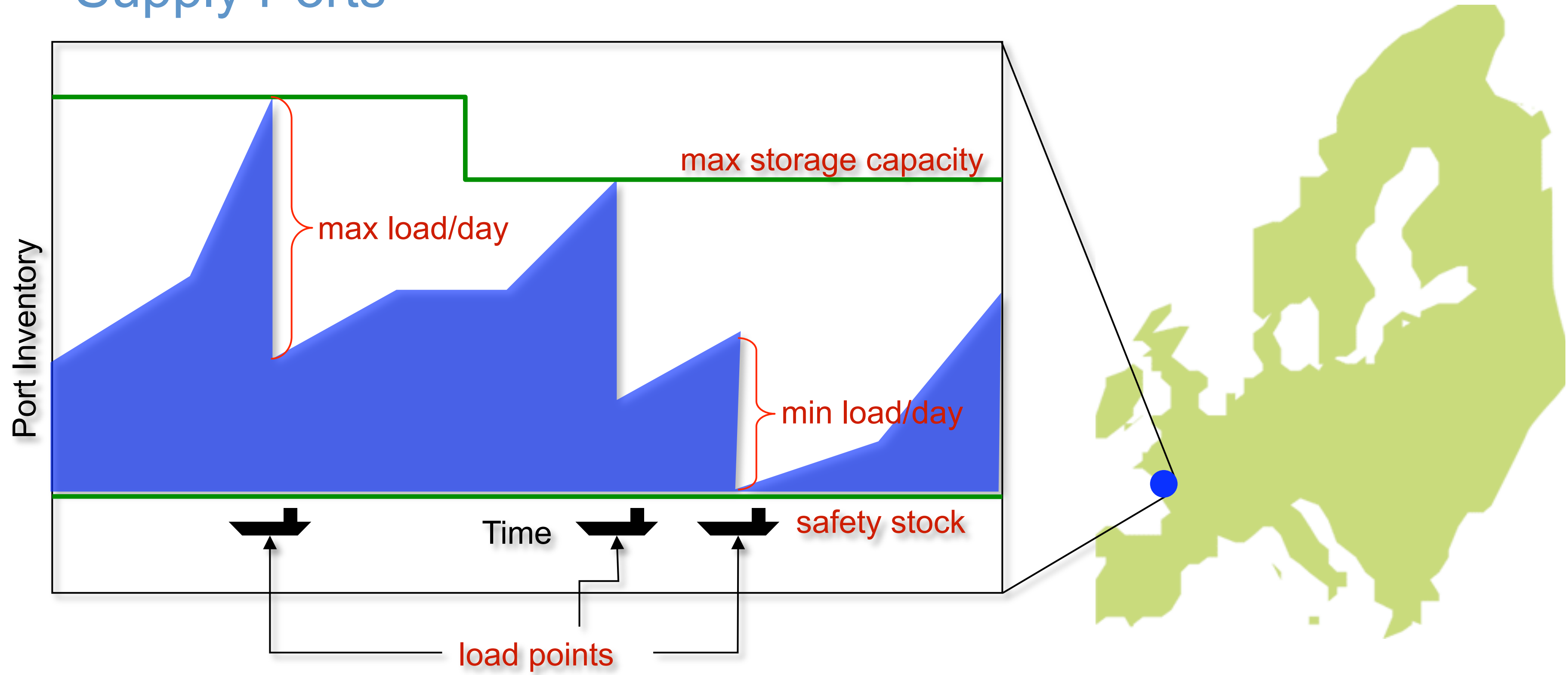
Maritime Inventory Routing

Maritime Inventory Routing



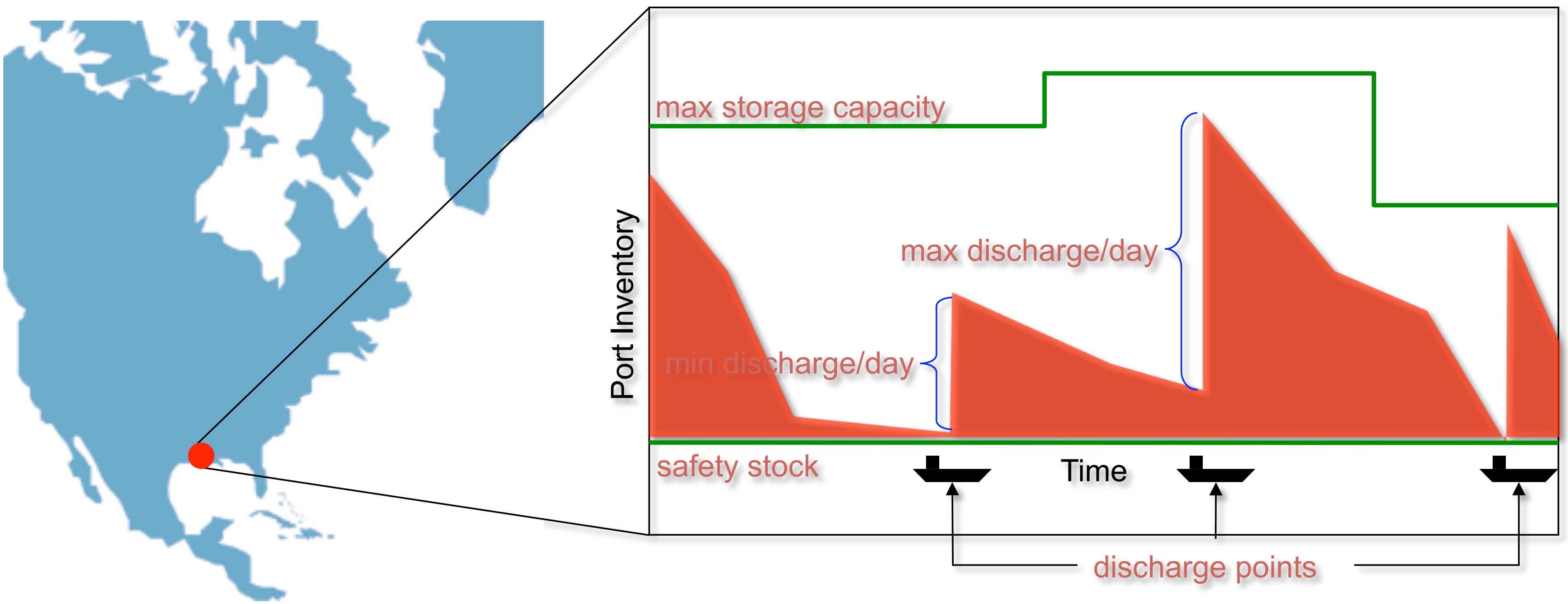
Maritime Inventory Routing

Supply Ports

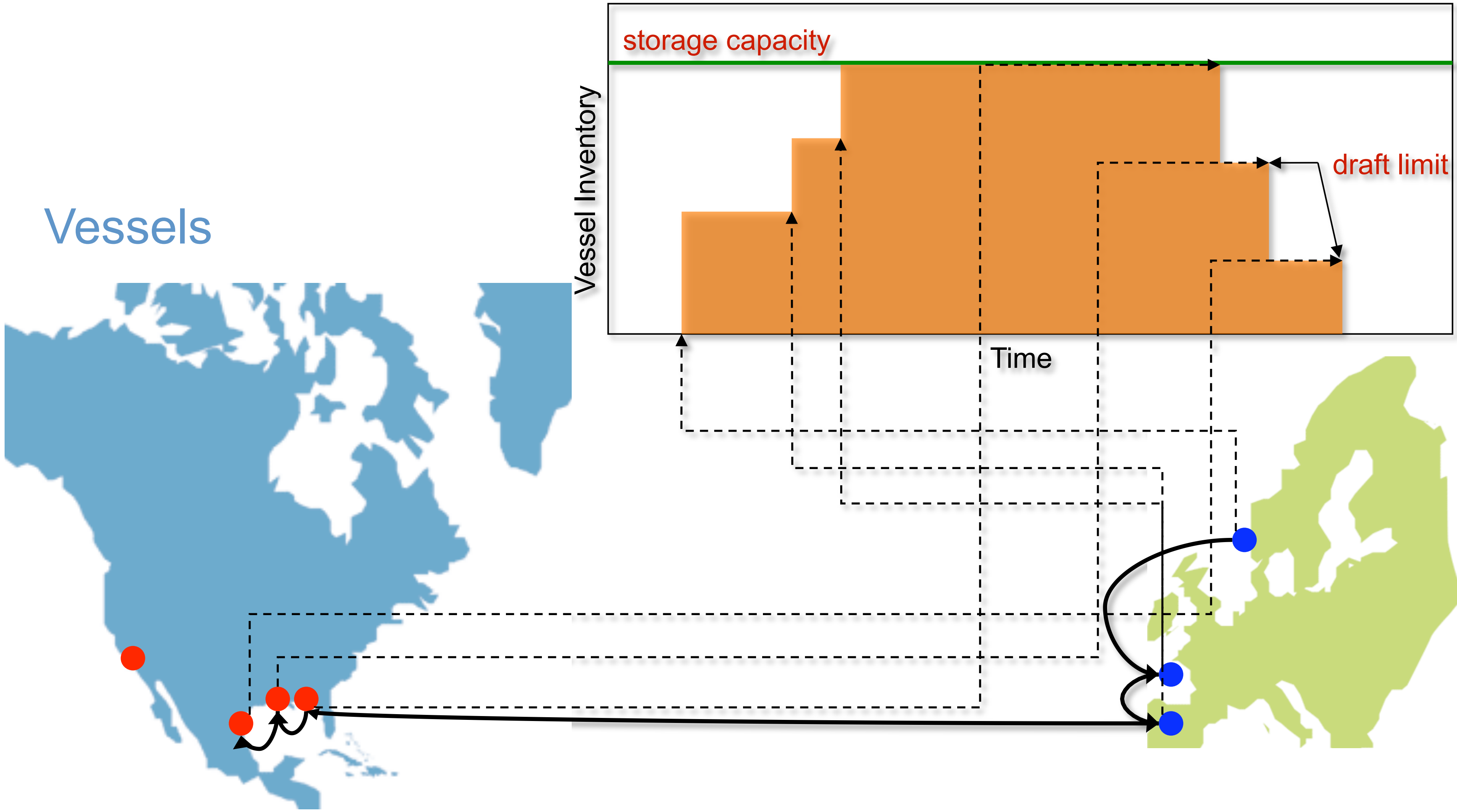


Maritime Inventory Routing

Consumption Ports



Maritime Inventory Routing



Maritime Inventory Routing - Formulation

$$\text{maximize} \boxed{\sum_{k} \sum_{(s,d)} p_s f^k_{(s,d)}} - \boxed{\sum_{k} \sum_{a \in A} c_a x^k_a}$$ Vessel transportation cost

Revenue from discharging product - cost of picking up product

$$\sum_{a \in \delta^+(s,d)} x^k_a - \sum_{a \in \delta^-(s,d)} x^k_a = \delta_{k,(s,d)}$$ Vessel balance in time-space network

$$y^k_{(s,d)} \leq \sum_{a \in \delta^-(s,d)} x^k_a$$ If vessel loads/discharges at (s,d) then must visit

$$\sum_{k} y_{(s,d)} \leq 1$$ Single vessel may load/discharge at a port per day

$$F^{min}_j y^k_{(s,d)} \leq f^k_{(s,d)} \leq F^{max}_j y^k_{(s,d)}$$ Amount loaded/discharged at (s,d) must fall within range

$$I^k_{d-1} + \sum_{s} f^k_{(s,d)} = I^k_d$$ Inventory balance on vessel

Routing variables binary

$$I_{s,d-1} + \sum_{k} f^k_{(s,d)} + R_{s,d} = I_{s,d}$$ Inventory balance at port

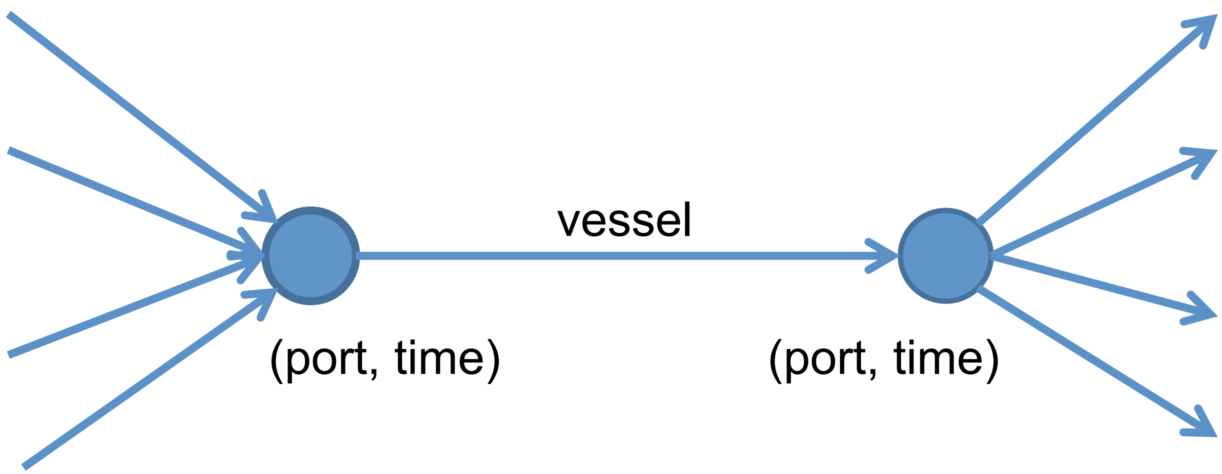
Load/discharge variables continuous

$$I^{min}_s \leq I_{s,d} \leq I^{max}_s$$ Inventory bounds at port

$$I^{min}_k \leq I^k_d \leq I^{max}_k$$ Inventory bounds at vessel

Problem-specific fixing and unfixing

PARTIAL PATH FIXING/UNFIXING



When fixing flow on the arc, immediately fix integral flows on outflow and inflow arcs at head and tail nodes (exploit flow-balance constraints)

Computational Study

Restrict-and-Relax Search

- Initial restricted IP from feasible solution: fix |V|-2 vessel routes
- Unfix if necessary, only at nodes that would be pruned by bound or infeasibility
- Fix if necessary, at a node if current LP is feasible and there are "not enough" fixed variables: 3*|V| arcs
- In a trial, unfix at most |V| partial paths
- In a trial, fix at most |V| partial paths
- At a node, try relaxing at most 5 times
- At a node, try fixing at most once
- Choose the partial paths to fix/unfix based on cumulative reduced costs

Time limit: 2000 seconds

Computational Results

Instance	Original IP	Restrict-and-Relax
mip-6-4-3-6	-5413.11	-5977.534
mip-6-4-3-7	-3279.32	-5912.800
mip-6-4-3-8	-5153.07	-5847.596
mip-6-4-3-9	-2182.03	-5180.212
mip-6-4-4-6	-5035.67	-4793.510
mip-6-4-4-7	-4471.51	-5551.830
mip-6-4-6-2	-4806.39	-7224.435
mip-6-6-4-3	-2395.84	-4502.810
mip-6-6-4-7	-2606.29	-5674.462
mip-6-6-4-9	-4600.78	-7541.257

Default solver and Restrict-and-Relax given same information and same amount of time

Questions?