Resource Constrained Shortest Path with a Super Additive Objective Function

Stefano Gualandi
Università di Pavia, Dipartimento di Matematica
Federico Malucelli
Politecnico di Milano, Dipartimento di Elettronica e Informazione

October 5, 2012

Introduction

Let $G = (N \cup \{s, t\}, A)$ be an (acyclic) digraph with source s and destination t. Every arc has a **weight** w_e and a **time** t_e.

Let K be a set of resources and r_e^k is the consumption of resource k on arc $e \in A$.

A path P_{st} from s to t is **resource feasible** iff at destination:

$$r^k(P_{st}) = \sum_{e \in P_{st}} r_e^k \leq U^k, \quad \forall k \in K$$

Problem (RCSP)

The Resource Constrained Shortest Path Problem consists in finding a **resource feasible** *path in G from s to t of* **minimum cost**.

Super Additivity

Definition (Path Super Additivity)

A (path) cost function is super additive iff:

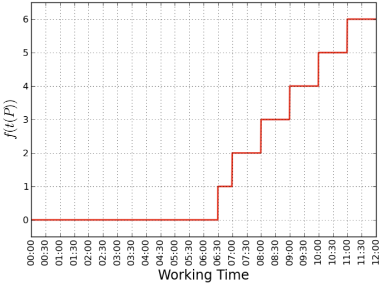
$$c(P_1 \cup P_2) \geq c(P_1) + c(P_2) \quad (1)$$

We consider here a specific type of super additive cost function:

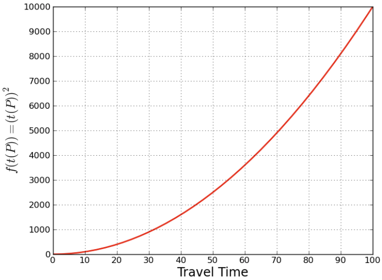
$$\begin{aligned} c(P) &= w(P) + f\Big(t(P)\Big) \\ &= \sum_{e \in P} w_e + f\Big(\sum_{e \in P} t_e\Big) \end{aligned}$$

where $f(\cdot)$ is a super additive function. Since $w(P)$ is additive, $c(P)$ is also super additive.

Examples: Stepwise and Quadratic Cost Functions



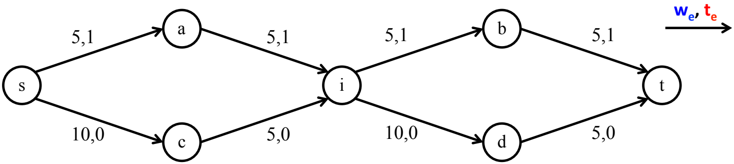
The extra allowances paid to bus drivers of an Italian trasportation company follow a stepwise cost function



"People value time nonlinearly: small amounts of time have relatively low value whereas large amounts of time are very valuable"
(Gabriel and Bernstein, 1997)

Bellmann's optimality conditions

Super additivity invalidates Bellmann's optimality conditions:
Two subpaths of an optimal path might be not optimal.



Example: Consider $c(P) = w(P) + f(P)$, with $f(t(P)) = (\sum_{e \in P} t_e)^2$
There are 4 paths:

$$P_1 = \{s, a, i, b, t\},\ w(P_1) = 20,\ f(P_1) = 16,\ c(P_1) = 36$$

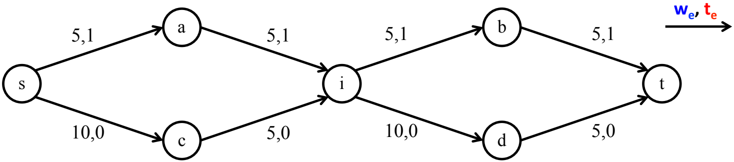
$$P_2 = \{s, c, i, b, t\},\ w(P_2) = 25,\ f(P_2) = 4,\ c(P_2) = 29$$

$$P_3 = \{s, a, i, d, t\},\ w(P_3) = 25,\ f(P_3) = 4,\ c(P_3) = 29$$

$$P_4 = \{s, c, i, d, t\},\ w(P_4) = 30,\ f(P_4) = 0,\ c(P_4) = 30$$

Bellmann's optimality conditions

Super additivity invalidates Bellmann's optimality conditions:
Two subpaths of an optimal path might be not optimal.



Example: Consider $c(P) = w(P) + f(P)$, with $f(t(P)) = (\sum_{e \in P} t_e)^2$
The optimal path $P_2 = \{s, c, i, b, t\}$ is composed of two subpaths:

$$P_{si} = \{s, c, i\} \text{ with cost } c(P_{si}) = 15 > 14 = c(\{s, a, i\})$$

$$P_{it} = \{i, b, t\} \text{ with cost } c(P_{it}) = 14$$

Remark: In addition our problem has bounded resources (. . . and side constraints)!

Our approach

We apply both resource-based and cost-based filtering algorithms to remove nodes and arcs as much as possible. **At the same time**, we keep on **updating lower and upper bounds** (FilterAndDive). When updating upper bounds, we can check additional **side constraints**.

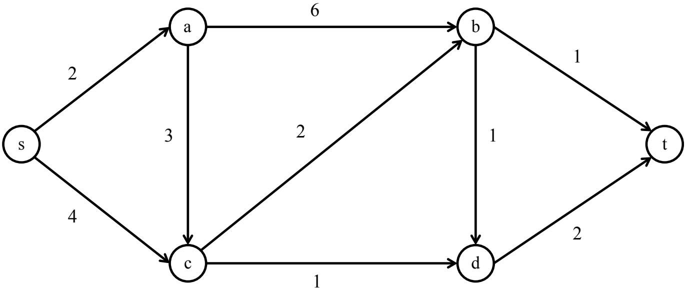
After that propagation reaches a fix point, we apply a near shortest path enumeration algorithm.

Resource-based Filtering

(Beasley and Christofides, 1989; Dumitrescu and Boland, 2003; Sellmann et al., 2007)

if $r^k(P^*_{si}) + r^k_e + r^k(P^*_{jt}) > U^k$ **then** remove arc $e = (i, j)$
where P^*_{si} and P^*_{jt} are shortest (k-th resource) paths.

Resource consumption of each arc. Upper resource bound $U = 7$.

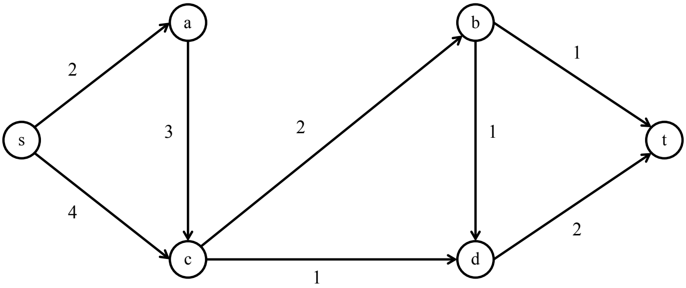


Resource-based Filtering

(Beasley and Christofides, 1989; Dumitrescu and Boland, 2003; Sellmann et al., 2007)

if $r^k(P^*_{si}) + r^k_e + r^k(P^*_{jt}) > U^k$ **then** remove arc $e = (i, j)$ where P^*_{si} and P^*_{jt} are shortest (k-th resource) paths.

Resource consumption of each arc. Upper resource bound $U = 7$.

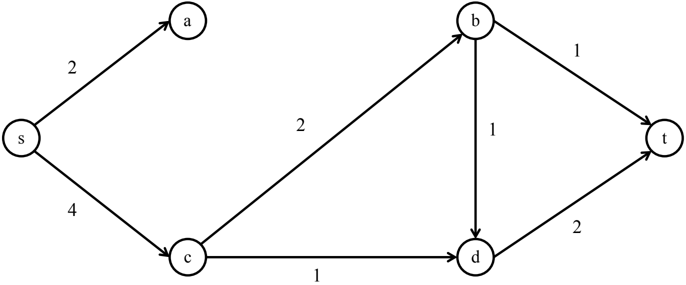


Resource-based Filtering

(Beasley and Christofides, 1989; Dumitrescu and Boland, 2003; Sellmann et al., 2007)

if $r^k(P^*_{si}) + r^k_e + r^k(P^*_{jt}) > U^k$ **then** remove arc $e = (i, j)$
where P^*_{si} and P^*_{jt} are shortest (k-th resource) paths.

Resource consumption of each arc. Upper resource bound $U = 7$.

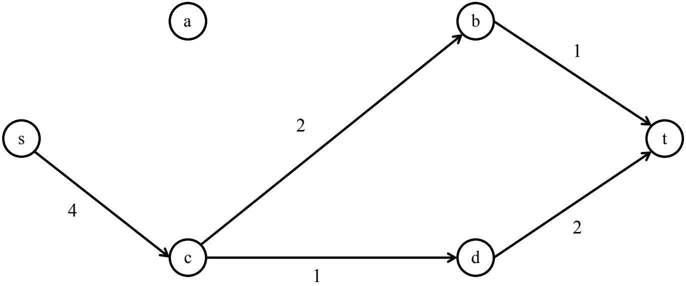


Resource-based Filtering

(Beasley and Christofides, 1989; Dumitrescu and Boland, 2003; Sellmann et al., 2007)

if $r^k(P^*_{si}) + r^k_e + r^k(P^*_{jt}) > U^k$ **then** remove arc $e = (i, j)$
where P^*_{si} and P^*_{jt} are shortest (k-th resource) paths.

Resource consumption of each arc. Upper resource bound $U = 7$.

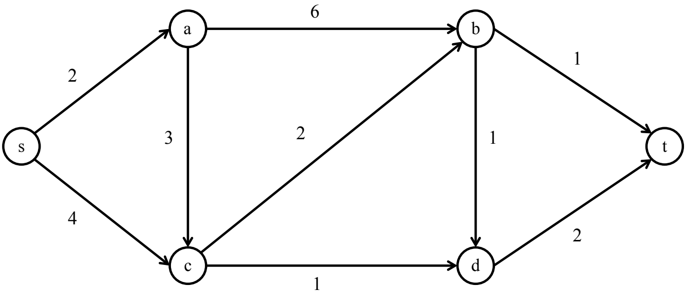


(Linear) Cost-based Filtering

(Beasley and Christofides, 1989; Dumitrescu and Boland, 2003; Sellmann et al., 2007)

if $w(P^*_{si}) + w_e + w(P^*_{jt}) > UB$ **then** remove arc $e = (i, j)$
where P^*_{si} and P^*_{jt} are shortest (weighted) paths.

Weight of each arc. Upper bound $UB = 7$.



Cost-based Filtering

if $LB(c(P^*_{s \xrightarrow{e} t})) \geq UB$ **then** remove arc e
where $P^*_{s \xrightarrow{e} t}$ is a shortest path from s to t via arc e.

There are at least three methods to compute such lower bound (see our poster!)

The most effective is based on a **Lagrangian Relaxation**

Lagrangian Relaxation: Arc-Flow Formulation

Iti is possible to formulate the following Lagrangian dual:

$$\Phi(\alpha, \beta) = -\sum_{k \in K} \alpha_k U^k +$$

$$+ \min \sum_{e \in A} \left(w_e + \sum_{k \in K} \alpha_k r_e^k + \beta t_e \right) x_e + f(z) - \beta z$$

$$\text{s.t.} \quad \sum_{e \in \delta_i^+} x_e - \sum_{e \in \delta_i^-} x_e = b_i \qquad \forall i \in N$$

$$x_e \geq 0 \qquad \forall e \in A.$$

This problem decomposes into two subproblems and is solved via a **subgradient optimization algorithm**:

1. The x variables define a *shortest path problem*
2. The z variable defines an *unconstrained optimization problem*

Cost-based Filtering

if $LB(c(P^*_{s \xrightarrow{e} t})) \geq UB$ **then** remove arc e
where $P^*_{s \xrightarrow{e} t}$ is a shortest path from s to t via arc e.

There are at least three methods to compute such lower bound (see our poster!)

The most effective is based on a **Lagrangian Relaxation**

$$c(P^*_{s \xrightarrow{e} t}) \geq \bar{w}(P^*_{s \xrightarrow{e} t}) + \min\{f(z) - \bar{\beta} z\}$$

[with reduced costs $\bar{w}_e = w_e + \sum_{k \in K} \bar{\alpha}_k r_e^k + \bar{\beta} t_e$]

Filter and Dive

Algorithm 1: FilterAndDive(G, LB, UB, F^g, B^g, U^g)

Input: $G = (N, A)$ directed graph and distance function $g(\cdot)$
Input: (LB, UB) lower and upper bounds on the optimal path
Input: F^g, B^g forward and backward shortest path tree as function of $g(\cdot)$
Input: U^g upper bound on the path length as function of $g(\cdot)$
Output: An optimum path, or updated UB, or a reduced graph

- **foreach** $i \in N$ **do**
  - **if** $F_i^g + B_i^g > U^g$ **then**
    - $N \leftarrow N \setminus \{i\}$
  - **else**
    - **foreach** $e = (i, j) \in A$ **do**
      - **if** $F_i^g + g(e) + B_j^g > U^g$ **then**
        - $A \leftarrow A \setminus \{e\}$
      - **else**
        - **if** PathCost(F_i^g, e, B_j^g) $< UB \wedge$ PathFeasible(F_i^g, e, B_j^g) **then**
          - $P_{st}^* \leftarrow$MakePath(F_i^g, e, B_j^g);
          - Update UB and store P_{st}^*;
          - **if** $LB \geq UB$ **then**
            - **return** P_{st}^* (that is an optimum path)
          - **else**
            - $A \leftarrow A \setminus \{e\}$

Closing the Duality Gap

After reaching a fix point, if $LB < UB$ then, we apply a **near shortest path** enumeration algorithm (Carlyle et al., 2008).

We compute shortest reversed distances for every resource and for reduced costs

Then we perform a depth-first search from s. When a vertex i is visited, the algorithm backtracks if

1. for any resource k, the consumption of P_{si} plus the reversed (resource) distance to t exceeds U^k
2. the reduced cost of P_{si} plus the reversed (reduced cost) distance to t exceeds UB
3. the cost $c(P_{si}) \geq UB$

Computational Results: Stepwise Function

Comparison of filtering algorithms for *real life instances*: the super additive function computes the **extra allowances** due to bus drivers.

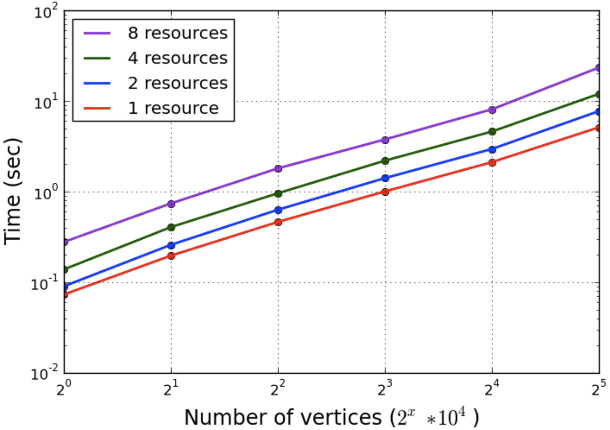
Each row gives the averages over 16 instances, with 7 resources.

- Δ is percentage of removed arcs
- Gap is $\frac{UB-Opt}{Opt} \times 100$

Graphs		Resource		Reduced Cost			Exact
n	m	Time	Δ	Time	Δ	Gap	Time
4137	135506	0.77	22.5%	3.12	30.2%	0.0%	75.1
2835	132468	0.59	40.3%	2.35	45.4%	0.0%	30.6
3792	134701	0.92	30.2%	2.87	37.4%	0.0%	69.3

Computational Results: $f(t(P)) = (t(P))^2$

Time to compute **optimal solutions**. DIMACS shortest path challenge instances (acyclic graphs). Average over 10 instances per type. The biggest instances have 320.000 nodes and 1.280.000 arcs.



Conclusions

- We have developed and implemented a Constrained Path Solver that handles super additive cost functions
- The cost-based filtering algorithm is very general and it could be implemented within a CP solver
- We are studying an alternative Lagrangian relaxation of the problem in order to get stronger lower bounds

Thanks for your attention!

Lagrangian Relaxation: Arc-Flow Formulation

The arc-flow LP relaxation of RCSP with a super additive cost function $f(\cdot)$ is:

$$\begin{aligned}
\min \quad & \sum_{e \in A} w_e x_e + f\left(\sum_{e \in A} t_e x_e \right) & \\
\text{s.t.} \quad & \sum_{e \in \delta_i^+} x_e - \sum_{e \in \delta_i^-} x_e = b_i = \begin{cases} +1 & \text{if } i = s \\ -1 & \text{if } i = t \\ 0 & \text{otherwise} \end{cases} & \forall i \in N \\
& \sum_{e \in A} r_e^k x_e \leq U^k & \forall k \in K \\
& x_e \geq 0 & \forall e \in A.
\end{aligned}$$

Lagrangian Relaxation: Arc-Flow Formulation

The arc-flow LP relaxation of RCSP with a super additive cost function $f(\cdot)$ is:

$$\min \sum_{e \in A} w_e x_e + f(z) \quad (2)$$

$$\text{s.t.} \sum_{e \in \delta_i^+} x_e - \sum_{e \in \delta_i^-} x_e = b_i \quad \forall i \in N \quad (3)$$

multiplier $\alpha_k \leq 0$ $\rightarrow$ $$\sum_{e \in A} r_e^k x_e \leq U^k \quad \forall k \in K \quad (4)$$

multiplier $\beta \leq 0$ $\rightarrow$ $$\sum_{e \in A} t_e x_e \leq z \quad (5)$$

$$x_e \geq 0 \quad \forall e \in A. \quad (6)$$

*Bibliography

JE Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19(4):379–394, 1989.

W.M. Carlyle, J.O. Royset, and R.K. Wood. Lagrangian relaxation and enumeration for solving constrained shortest-path problems. *Networks*, 52 (4):256–270, 2008.

I. Dumitrescu and N. Boland. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problema. *Networks*, 42 (3):135–153, 2003.

S.A. Gabriel and D. Bernstein. The traffic equilibrium problem with nonadditive path costs. *Transportation Science*, 31(4):337–348, 1997.

M. Sellmann, T. Gellermann, and R. Wright. Cost-based filtering for shorter path constraints. *Constraints*, 12:207–238, 2007.

G. Tsaggouris and C. Zaroliagis. Non-additive shortest paths. *European Symposium on Algorithms*, pages 822–834, 2004.