

## Multicriteria Optimization

### Solution Sheet 9

#### IN-CLASS EXERCISES

(To be done in the tutorial on December 20th, 2011)

#### Problem 9.1 – SUM-MAX-SPP

Let  $G = (V, A)$  be a directed graph,  $s, t \in V$  and denote by  $\mathcal{P}^{s,t}$  the set of all  $s$ - $t$  paths in  $G$ , where a path  $P \in \mathcal{P}^{s,t}$  is defined by its edges. Consider the following shortest path problem with one sum and  $p - 1$  bottleneck objectives:

$$\min \left( \sum_{a \in P} c^1(a), \max_{a \in P} c^2(a), \dots, \max_{a \in P} c^p(a) \right) \quad (\text{SUM-MAX-SPP})$$

s. t.  $P \in \mathcal{P}^{s,t}$ .

- Find a good upper bound on the number of nondominated points for SUM-MAX-SPP.
- Find an algorithm to find a complete minimal set of efficient solutions. You do not have to give a rigorous proof of the correctness. *Hint: Try to first find a solution for  $p = 2$ , then generalize.*
- How can we solve the problem if there are bottleneck objectives only?

#### Solution 9.1:

For a path  $P \in \mathcal{P}^{s,t}$ , we define  $c^1(P) = \sum_{a \in P} c^1(a)$  and  $c^j(P) = \max_{a \in P} c^j(a)$  for  $j = 2, \dots, p$ . For  $\hat{c} \in \mathbb{R}^{p-1}$ , we define

$$\mathcal{P}_{\hat{c}}^{s,t} = \left\{ P \in \mathcal{P}^{s,t} : c^j(P) \leq \hat{c}_{j-1} \text{ for } j = 2, \dots, p \right\}.$$

- Let  $\mathcal{X}'$  be a minimal complete set of efficient solutions. For each  $P \in \mathcal{X}'$ , it must hold that

$$c^1(P) = \min \left\{ c^1(P') : P' \in \mathcal{P}^{s,t} \text{ with } c^2(P') \leq c^2(P), \dots, c^p(P') \leq c^p(P) \right\},$$

i. e.,  $P$  must be a minimizer of  $c^1$  within  $\mathcal{P}_{(c^2(P), \dots, c^p(P))}^{s,t}$ . Otherwise, if there were a  $P'$  with  $c^1(P') \leq c^1(P)$  satisfying all the bottlenecks defined by  $P$ , that path would dominate  $P$ . Thus each configuration of bottleneck values contributes at most one element to  $\mathcal{X}'$ .

Let

$$C^j = \{v \in \mathbb{R} : \exists a \in A \text{ with } c^j(a) = v\}$$

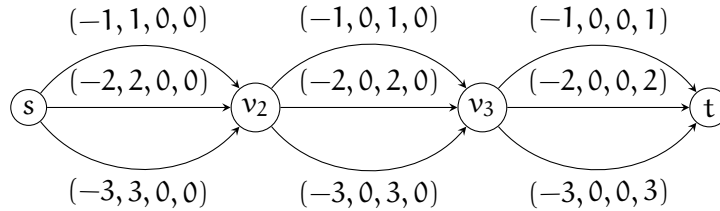
be the set of distinct cost values of the  $j$ -th objective function. Then the set of possible configurations of the bottleneck objectives is

$$C^1 \times C^2 \times \dots \times C^p$$

and thus  $\prod_{j=2}^p |C^j|$  is an upper bound for  $|\mathcal{X}'| = |\mathcal{Y}_N|$ . Note that  $|C^j| \leq |A|$ , so a weaker bound not involving the  $C^j$  would be  $|\mathcal{Y}_N| \leq |A|^{p-1}$ .

We now give an infinite class of graphs for which the first bound is tight. Assume  $|A| = k \cdot (p-1)$  for some  $k \in \mathbb{N}$  and define  $G = (V, A)$  as a line graph with parallel arcs as follows:

- $V = (v_1 = s, v_2, \dots, v_p = t)$
- For each  $i \in \{2, \dots, p\}$  there are  $k$  parallel arcs  $(v_{i-1}, v_i)$  which we denote by  $a^{i,j}$  for  $j = 1, \dots, k$ .
- For an arc  $a^{i,j} \in A$ , the cost vector is defined by  $c^1(a) = -j$ ,  $c^i(a) = j$ ,  $c^l(a) = 0$  for  $l \notin \{1, i\}$ . An example for  $p = 4$  and  $k = 3$  is given below.



We claim that  $\mathcal{P}^{s,t}$  is a minimal efficient set, i.e. all paths are efficient and have different objective values. As for the latter, observe that the choice of the  $i$ -th arc of a path,  $a^{i,j}$ , uniquely determines  $c^{i+1}(P)$ , because  $a^{i,j}$  is the only arc  $a \in P$  with  $c^{i+1}(a) \neq 0$ . Because  $c^{i+1}(a^{i,j})$  is different for all  $j = 1, \dots, k$ , all paths have different costs in the bottleneck objectives.

To show that all paths are efficient, assume for the contrary that a path  $P$  is dominated by some other path  $\hat{P} \neq P$ . Let  $i$  be minimal such that  $P \ni a^{i,j} \neq a^{i,\hat{j}} \in \hat{P}$ , i.e. the paths differ in the  $i-1$ -st arc. Since  $\hat{P}$  dominates  $P$ , we must have  $\hat{j} < j$ , otherwise  $c^i(P) = j < c^i(\hat{P}) = \hat{j}$  and  $P$  would not be dominated.

But then  $c^1(a^{i,\hat{j}}) > c^1(a^{i,j})$ , so there must be at least one index  $i' > i$  where  $\hat{P}$  takes an arc with larger index:  $P \ni a^{i',j'} \neq a^{i',\hat{j}'} \in \hat{P}$  with  $\hat{j}' > j'$ . This however implies  $c^{i'}(P) < c^{i'}(\hat{P})$ , which is a contradiction to  $\hat{P}$  dominating  $P$ .

- b) We claim that Algorithm 1 computes a minimal efficient set for SUM-MAX-SPP. Observe that the **for** loop in Line 8 traverses all possible configurations of the bottleneck costs. Thus, by the argumentation of part a), the total set of paths generated in Line 9 contains a minimal complete efficient set. It is left to show that the detection of dominated solutions in the following lines is correct; i.e. that in Line 24, we add exactly one path for each  $y \in \mathcal{Y}_N$ .

**Lemma 1:** For  $\hat{c} \in C^2 \times \dots \times C^p$ , let

$$c^*(\hat{c}) = \min_{P \in \mathcal{P}_{\hat{c}}^{s,t}} c^1(P) \text{ and } P^*(\hat{c}) = \arg \min_{P \in \mathcal{P}_{\hat{c}}^{s,t}} c^1(P),$$

if such a path exists. Then the set

$$\mathcal{P}_{\text{Alg}} \{P^*(\hat{c}) : \mathcal{P}_{\hat{c}}^{s,t} \neq \emptyset, c^*(\hat{c}') > c^*(\hat{c}) \text{ for all } \hat{c}' \leq \hat{c}, \hat{c}' \neq \hat{c}\}$$

is a minimal complete set of efficient solutions for SUM-MAX-SPP.

**Proof.** Let  $P \in \mathcal{X}_E$  be an efficient path with objective value  $c(P) = c^1(P), \dots, c^p(P)$ . Let  $\hat{c} = (c^2(P), \dots, c^p(P))$ , then by part a),  $c(P) = c^*(\hat{c})$ . Assume that  $\mathcal{P}_{\text{Alg}}$  does not contain any path  $P'$  with  $c(P') = c(P)$ . Then by definition of  $\mathcal{P}_{\text{Alg}}$  there must be some  $\hat{c}' \leq \hat{c}$  such that  $c^*(\hat{c}') \leq c^*(\hat{c})$ . But this means that  $P^*(\hat{c}')$  dominates  $P$ , contradicting  $P \in \mathcal{X}_E$ .  $\square$

With the additional observation that for any  $\hat{c}' \geq \hat{c}$  we necessarily have  $c^*(\hat{c}') \leq c^*(\hat{c})$  (since constraints are relaxed) it is easy to see that Algorithm 1 computes  $\mathcal{P}_{\text{Alg}}$  and is thus correct.

Its running time is, if we omit output cost in Line 24,  $\mathcal{O}\left(\prod_{j=2}^p |C^j| (|V| \log |V| + |A|)\right)$ : The shortest path computation in Line 9 can be accomplished in  $\Theta(|V| \log |V| + |A|)$  using Dijkstra's algorithm with Fibonacci heaps. Disabling arcs that don't meet the bottleneck constraints  $\hat{c}$  takes only linear time for each such  $\hat{c}$ .

c) When there are bottleneck objectives only, we could use a modification of Algorithm 1 to compute a minimal complete efficient set. Instead of the minimum  $c^1$ -values for each bottleneck configuration, however, the Value array only needs to store a boolean flag, whether there exists a path in  $\mathcal{P}_{\hat{c}}^{s,t}$  or not. Thus in Line 9 we can use a linear-time connectedness test instead of shortest path computation. The following observations could be used to further reduce the running time:

- a path with objective value  $c(P) = c^1(P), \dots, c^p(P)$  is efficient if and only if there does not exist a path  $P'$  with  $c(P') \leq c(P)$ ,  $c(P') \neq c(P)$ ;
- if  $\mathcal{P}_{\hat{c}}^{s,t}$  is empty for some  $\hat{c}$ , then it is also empty for every  $\hat{c}' \leq \hat{c}$ ;
- if  $\mathcal{P}_{\hat{c}}^{s,t}$  is nonempty, that also holds for all  $\hat{c}' \geq \hat{c}$ .

---

**Algorithm 1** Algorithm solving SUM-MAX-SPP

---

```
1: Input: Graph  $G = (V, A)$  with  $p$  objective functions  $c^1, \dots, c^p$ .
2: Output: A minimal complete set  $S$  of efficient solutions for SUM-MAX-SPP.
3: for all  $j = 2, \dots, p$  do
4:   Compute  $C^j$  and sort it such that  $C_1^j \leq \dots \leq C_{|C_j|}^j$ 
5: end for
6: Allocate a  $p - 1$ -dimensional array  $\text{Value}[|C^2|, \dots, |C^p|]$ 
7:  $S \leftarrow \emptyset$ 
8: for all  $I \in [1, |C^2|] \times \dots \times [1, |C^p|]$ , traversed in lexicographic order do
9:    $\hat{c} \leftarrow (C_{I_1}^2, \dots, C_{I_{p-1}}^p)$ 
10:  Find a path  $P = \arg \min_{P \in \mathcal{P}_{\hat{c}}^{s,t}} c^1(P)$ 
11:  if there is no such path ( $\mathcal{P}_{\hat{c}}^{s,t}$  is empty) then
12:     $\text{Value}[I] \leftarrow \infty$ 
13:  else
14:     $\text{Value}[I] \leftarrow c^1(P)$ 
15:     $\text{dominated} \leftarrow \text{False}$ 
16:    for all  $i = 1, \dots, p - 1$  with  $I_i > 1$  do
17:       $\hat{I} \leftarrow I$ 
18:       $\hat{I}_i = \hat{I}_i - 1$ 
19:      if  $\text{Value}[\hat{I}] = c^1(P)$  then
20:         $\text{dominated} \leftarrow \text{True}$ 
21:        break
22:      end if
23:      if not  $\text{dominated}$  then
24:         $S \leftarrow S \cup \{P\}$ 
25:      end if
26:    end for
27:  end if
28: end for
29: return  $S$ 
```

---

**TURN-IN EXERCISES**

(Please hand in by January 5th, 2012)

**Problem 9.2**

- a) Using the multicriteria label-setting algorithm from the lecture, compute a minimal complete set of efficient paths from  $v_1$  to  $v_7$  in the graph of Figure 1.
- b) Consider a directed graph  $G = (V, A)$  with cost function  $c : A \rightarrow \mathbb{R}^p$  not containing negative cycles, i. e., for all cycles  $Z$  in  $G$  we have  $c(Z) = \sum_{a \in Z} c(a) \geq 0$ .
  1. Prove the following statement: Let  $P^{s,t}$  be an efficient  $s$ - $t$  path on  $G$ . Then any subpath  $P^{u,v}$  from  $u$  to  $v$ , where  $u$  and  $v$  are vertices on  $P^{s,t}$ , is an efficient path from  $u$  to  $v$ .
  2. Show that the converse is not true in general, i. e. compositions of efficient paths are not necessarily efficient.

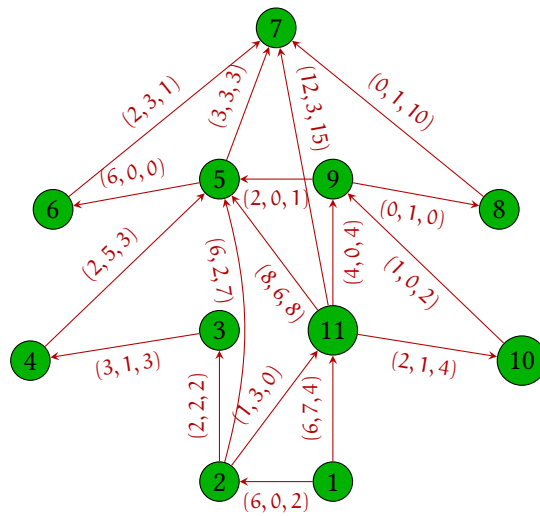


Figure 1: The graph for Problem 2

### Problem 9.3 – Multicriteria FLOYDWARSHALL

Recall how the algorithm of Floyd and Warshall computes all-to-all shortest paths in a directed graph under the assumption that it does not contain negative dicycles. The objective of this exercise is to extend that algorithm so as to find a minimal complete efficient set for a multicriteria shortest path problem (which also must not contain any negative cycle in the sense of Problem 2).

To solve this exercise, assign a list of labels to each node, similar as for the multicriteria label-setting algorithm in the lecture. Then use the generalized Bellman principle of the previous exercise to restrict the set of possible efficient paths in the inner FOR loop of FLOYDWARSHALL.

Download of exercises at  
<http://optimierung.mathematik.uni-kl.de/teaching/ws1112/multicriteria-opt.html>