Knapsack problems with special neighbor constraints on directed co-graphs

Steffen Goebbels [1], Frank Gurski [2], Dominique Komander [2]

[1]Niederrhein University of Applied Sciences
Faculty of Electrical Engineering and Computer Science, D-47805 Krefeld
[2]Heinrich Heine University
Institute of Computer Science, D-40225 Düsseldorf

OR 2021

HEINRICH HEINE UNIVERSITÄT DÜSSELDORF

Hochschule Niederrhein
University of Applied Sciences

iPattern
Institut für Mustererkennung
Institute for Pattern Recognition

Outline

- **The knapsack problem, directed graphs and graph constraints**
- Directed co-graph constraints
  - Uniform all-neighbors problem on directed co-graphs
  - General one-neighbor problem on directed co-graphs
- Problems on msp-digraphs and directed trees

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Hochschule Niederrhein
University of Applied Sciences

iPattern
Institut für Mustererkennung
Institute for Pattern Recognition

The knapsack problem (KP)

- A set $A = \{a_1, \ldots, a_n\}$ of n items is given.
- Every item a_j has a size s_j and a profit $p_j \in \mathbb{N}_0 := \{0, 1, \ldots\}$.

The task is to choose a subset A' of A, such that the profit

$$p(A') := \sum_{a_j \in A'} p_j$$

is maximized subject to

$$s(A') := \sum_{a_j \in A'} s_j \leq c,$$

where $c \in \mathbb{N}_0$ is a given capacity. With dynamic programming, NP-hard KP can be solved in $\mathcal{O}(nc)$ and in $\mathcal{O}(nP)$ time,

$$P := \sum_{j=1}^{n} p_j \leq n \cdot \max_{1 \leq j \leq n} p_j.$$

Directed graphs

A **directed graph** or **digraph** is a pair $G = (A, E)$, where A is a finite set of **vertices** (the items of KP) and

$$E \subseteq \{(u, v) \mid u, v \in A,\ u \neq v\}$$

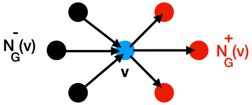
is a finite set of ordered pairs of distinct vertices called **arcs** or **directed edges**.

Directed graphs (2)

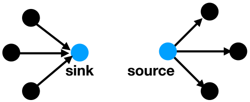
For a vertex $v \in A$, the sets

$$N_G^+(v) = \{u \in A \mid (v,u) \in E\} \text{ and } N_G^-(v) = \{u \in A \mid (u,v) \in E\}$$

are called the **set of all successors** and the **set of all predecessors** of v in G, respectively.



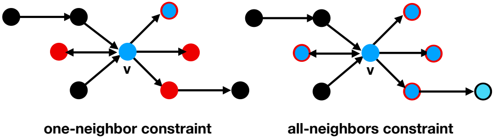
A vertex v is called a **sink** iff $N_G^+(v) = \emptyset$. It is called a **source** iff $N_G^-(v) = \emptyset$.



Directed graph constraints

- **One-neighbor constraint**[1]: An item $v \in A$ can be chosen into $A' \subseteq A$ only if at least one of its successors in $N_G^+(v)$ is chosen, i.e.,
$$\left(v \in A' \wedge N_G^+(v) \neq \emptyset\right) \Rightarrow N_G^+(v) \cap A' \neq \emptyset.$$

- **All-neighbors constraint**: An item $v \in A$ can be chosen into $A' \subseteq A$ only if all its successors in $N_G^+(v)$ are chosen, i.e.,
$$v \in A' \Rightarrow N_G^+(v) \subseteq A'.$$



[1] Borradaile, G., Heeringa, B., Wilfong, G.: The 1-neighbour knapsack problem. In: Proc. IWOCA, LNCS, vol. 7056, pp. 71–84 (2011)

HEINRICH HEINE UNIVERSITÄT DÜSSELDORF
Hochschule Niederrhein University of Applied Sciences
iPattern Institut für Mustererkennung Institute for Pattern Recognition

Time complexity of knapsack problems with neighbor constraints on graphs

Known results:[2]

	graph	one-neighbor	all-neighbors
uniform, i.e.	undirected[3]	linear	$\mathcal{O}(n \cdot c) \subseteq \mathcal{O}(n^2)$
$s_j = p_j = 1$	directed	strongly NP-hard	strongly NP-hard
general	undirected	APX-hard	PFTAS
	directed	strongly NP-hard	strongly NP-hard

[2] Borradaile, G., Heeringa, B., Wilfong, G.: The knapsack problem with neighbour constraints. Journal of Discrete Algorithms 16, 224–235 (2012)

[3] Undirected edges can be interpreted as opposite edges in a directed graph.

Outline

- The knapsack problem, directed graphs and graph constraints
- **Directed co-graph constraints**
  - Uniform all-neighbors problem on directed co-graphs
  - General one-neighbor problem on directed co-graphs
- Problems on msp-digraphs and directed trees

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Hochschule Niederrhein
University of Applied Sciences

iPattern
Institut für Mustererkennung
Institute for Pattern Recognition

Directed co-graphs

The class of **directed co-graphs** is recursively defined as follows.

(i) Every digraph on a single vertex $(\{v\}, \emptyset)$ is a directed co-graph.

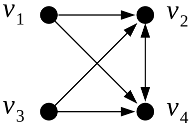
(ii) If $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are vertex-disjoint directed co-graphs, then directed co-graphs are

1. the **disjoint union** $G_1 \oplus G_2$ with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$,
2. the **order composition** $G_1 \oslash G_2$, defined by their disjoint union plus all possible edges only directed from V_1 to V_2, and
3. the **series composition** $G_1 \otimes G_2$, defined by their disjoint union plus all possible edges between V_1 and V_2 in both directions.

Directed co-graphs (2)

- Every expression X using the operations $\oplus$, $\oslash$, and $\otimes$ is called a **di-co-expression** and digraph(X) is the represented graph with $|X|$ vertices.
- The expression defines the **di-co-tree** T, the leaves are the vertices of digraph(X) and the inner nodes are the operations.
- For a subtree of T rooted at u, let $X(u)$ be the corresponding sub-expression of X.

Example: $X = (v_1 \oplus v_3) \oslash (v_2 \otimes v_4)$ defines this digraph(X):



Time complexity of knapsack problems with neighbor constraints on directed co-graphs

	one-neighbor	all-neighbors
uniform	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$
general	$\mathcal{O}(nP^2 + n^2)$	$\mathcal{O}(n(P+1)\max\{n, P+1\})$

HEINRICH HEINE UNIVERSITÄT DÜSSELDORF
Hochschule Niederrhein University of Applied Sciences
iPattern Institut für Mustererkennung Institute for Pattern Recognition

Outline

- The knapsack problem, directed graphs and graph constraints
- Directed co-graph constraints
  - **Uniform all-neighbors problem on directed co-graphs**
  - General one-neighbor problem on directed co-graphs
- Problems on msp-digraphs and directed trees

Uniform all-neighbors problem on directed co-graphs

For every vertex u of T and integer $0 \leq p \leq c \leq n$ we compute

$$F(X(u),p) := \begin{cases} 1: & \text{a feasible solution with profit } p \text{ exists in digraph}(X(u)) \\ 0: & \text{otherwise.} \end{cases}$$

$F(X(u), p) := 0$

if $p = 0 \vee p > \min\{|X(u)|, c\}$ **then**
  if $p = 0$ **then** $F(X(u), p) := 1$

else if $|X(u)| = 1$ **then**
  if $p = 1$ **then** $F(X(u), p) := 1$

else if $X(u) = L \oplus R$ **then**
  for $p' := \max\{0, p - |R|\}$; $p' \leq \min\{p, |L|\} \wedge F(X(u), p) = 0$; $p' := p' + 1$ **do**
    if $F(L, p') = 1 \wedge F(R, p - p') = 1$ **then** $F(X(u), p) := 1$;

else if $X(u) = L \oslash R$ **then**
  if $|R| \geq p$ **then**
    if $F(R, p) = 1$ **then** $F(X(u), p) := 1$
  else if $F(L, p - |R|) = 1$ **then** $F(X(u), p) := 1$

else if $X = L \otimes R$ **then**
  if $p = |X(u)|$ **then** $F(X(u), p) := 1$

Outline

- The knapsack problem, directed graphs and graph constraints
- Directed co-graph constraints
  - Uniform all-neighbors problem on directed co-graphs
  - **General one-neighbor problem on directed co-graphs**
- Problems on msp-digraphs and directed trees

General one-neighbor problem on directed co-graphs

Let $F(X, p, k)$ be the minimum size of a solution fulfilling the one-neighbor constraint with profit exactly p in digraph(X) that

- contains a sink if $k = 1$
- and does not possess any sinks if $k = 0$.

We set $F(X, p, k)$ to ∞, whenever there is no such solution.

The flag k is required for considering $X = L \oslash R$:
If a non-empty solution has all vertices in digraph(L) then it cannot have a sink because of the $\oslash$-operation and the neighbor condition.

HEINRICH HEINE UNIVERSITÄT DÜSSELDORF
Hochschule Niederrhein University of Applied Sciences
iPattern Institut für Mustererkennung Institute for Pattern Recognition

General one-neighbor problem (2)

Items are allowed to have zero profits. Non-empty zero profit solutions help to fulfill the neighbor constraint.
Let $\tilde{F}(X, p, k)$ be the minimum size of a **non-empty** solution fulfilling the one-neighbor constraint with profit exactly p in digraph(X) that contains a sink for $k = 1$ and does not contain a sink for $k = 0$. If there is no such solution, then $\tilde{F}(X, p, k) := \infty$.
Thus

$$F(X, p, k) := \begin{cases} \tilde{F}(X, p, k), & \text{if } p > 0 \vee k = 1, \\ 0, & \text{else.} \end{cases}$$

Outline

- The knapsack problem, directed graphs and graph constraints
- Directed co-graph constraints
  - Uniform all-neighbors problem on directed co-graphs
  - General one-neighbor problem on directed co-graphs
- **Problems on msp-digraphs and directed trees**

Minimal series-parallel digraphs (msp-digraphs)

Minimal series-parallel digraphs are recursively defined:

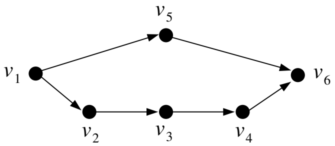
(i) Every digraph on a single vertex $(\{v\}, \emptyset)$ is an msp-digraph.

(ii) If $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are vertex-disjoint msp-digraphs, then msp-digraphs are

1. the **parallel composition** $G_1 \cup G_2 = (V_1 \cup V_2, E_1 \cup E_2)$,
2. the **series composition** $G_1 \times G_2 = (V_1 \cup V_2, E_1 \cup E_2 \cup (O_1 \times I_2))$, where O_1 is the set of sinks in G_1 and I_2 is the set of sources in G_2.

Example:

$$\Big(((\{v_1\}, \emptyset) \times ([(\{v_2\}, \emptyset) \times ((\{v_3\}, \emptyset) \times (\{v_4\}, \emptyset))] \cup (\{v_5\}, \emptyset))\Big) \times (\{v_6\}, \emptyset)$$



HEINRICH HEINE UNIVERSITÄT DÜSSELDORF

Hochschule Niederrhein University of Applied Sciences

iPattern Institut für Mustererkennung Institute for Pattern Recognition

Time complexity of knapsack problems with neighbor constraints on msp-digraphs

	one-neighbor	all-neighbors
uniform	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$
general	$\mathcal{O}(nP^2 + n^2)$	$\mathcal{O}(n(P+1)\max\{n, P+1\})$

Time complexity of knapsack problems with neighbor constraints on trees

A **directed tree** is a directed graph for which the underlying undirected graph is a tree. In directed trees, opposite edges are allowed.

	graph	one-neighbor	all-neighbors
uniform	undirected	linear[4]	$\mathcal{O}(1)$
	directed	$\mathcal{O}(n^3)$	$\mathcal{O}\left(n^3\right)$
general	undirected	NP-hard	$\mathcal{O}(n)$
	directed	NP-hard	NP-hard
		$\mathcal{O}(nP^2 + n)$	$\mathcal{O}(n(P+1)(P+n))$

[4] Borradaile, G., Heeringa, B., Wilfong, G.: The knapsack problem with neighbour constraints. Journal of Discrete Algorithms 16, 224–235 (2012)

Questions?

?

Backup: Fully Polynomial Time Approximation Scheme (FPTAS)

- If we exclude zero profits, the one-neighbor and all-neighbors problems become *subset selection problems*.
- Thus, shown pseudo-polynomial algorithms imply a FPTAS[5].

[5] Pruhs, K., Woeginger, G.: Approximation schemes for a class of subset selection problems. Theoretical Computer Science 382(2), 151–156 (2007)

Backup: General one-neighbor problem (3)

$\tilde{F}(X(u), p, k) := \infty$
if $|X(u)| = 1$ **then** let $a_i \in V$ be the only vertex.
  if $k = 1 \land p = p_i$ **then** $\tilde{F}(X(u), p, k) := s_i$
else if $X(u) = L \oplus R$ **then**
  ▷ Consider empty and non-empty solutions with zero profit in R or L
  $\tilde{F}(X(u), p, k) := \tilde{F}(L, p, k)$
  if $\tilde{F}(X(u), p, k) > \tilde{F}(R, p, k)$ **then** $\tilde{F}(X(u), p, k) := \tilde{F}(R, p, k)$
  ▷ Consider solutions with positive profit in L and R
  for $p' = 1$; $p' < p$; $p' := p' + 1$ **do**
    if $k = 0$ **then**
      $S := \tilde{F}(L, p', 0) + \tilde{F}(R, p - p', 0)$
      if $\tilde{F}(X(u), p, k) > S$ **then** $\tilde{F}(X(u), p, k) := S$
    else
      $S_1 := \tilde{F}(L, p', 1) + \tilde{F}(R, p - p', 0)$, $S_2 := \tilde{F}(L, p', 0) + \tilde{F}(R, p - p', 1)$
      $S_3 := \tilde{F}(L, p', 1) + \tilde{F}(R, p - p', 1)$
      for $i = 1$; $i \leq 3$; $i := i + 1$ **do**
        if $\tilde{F}(X(u), p, k) > S_i$ **then** $\tilde{F}(X(u), p, k) := S_i$
else if $X(u) = L \oslash R$ **then**
  if $k = 0$ **then** $\tilde{F}(X(u), p, k) := \tilde{F}(L, p, k)$
  if $\tilde{F}(X(u), p, k) > \tilde{F}(R, p, k)$ **then** $\tilde{F}(X(u), p, k) := \tilde{F}(R, p, k)$
  for $p'' = 0$; $p'' < p$; $p'' := p'' + 1$ **do**
    $S_R := \tilde{F}(R, p'', k)$
    if $S_R < \infty$ **then**
      $S_L := \hat{F}(L, p - p'')$
      if $\tilde{F}(X(u), p, k) > S_L + S_R$ **then** $\tilde{F}(X(u), p, k) := S_L + S_R$

Backup: General one-neighbor problem (4)

if $X(u) = L \otimes R$ **then** ▷ digraph($X(u)$) has no sinks
  if $k = 0$ **then**
    $\tilde{F}(X(u), p, k) := \tilde{F}(L, p, 0)$, $S_R := \tilde{F}(R, p, 0)$
    if $S_R < \tilde{F}(X(u), p, k)$ **then** $\tilde{F}(X(u), p, k) := S_R$
    ▷ Consider solutions with at least one vertex from both digraphs without restrictions
    for $p' = 0$; $p' \leq p$; $p' := p' + 1$ **do**
      $S_L := \hat{F}(L, p')$
      $S_R := \hat{F}(R, p - p')$
      if $S_L + S_R < \tilde{F}(X(u), p, k)$ **then** $\tilde{F}(X(u), p, k) := S_L + S_R$

The algorithm uses following dynamic program for knapsack on directed co-graphs without graph restrictions:

if $|X(u)| = 1$ **then** let $a_i \in V$ be the only vertex.
  if $p = p_i$ **then** $\hat{F}(X(u), p) := s_i$
  else $\hat{F}(X(u), p) := \infty$
else
  X has one of the representations $X = L \oplus R$, $X = L \oslash R$ or $X = L \otimes R$
  $\hat{F}(X(u), p) := \min\{\hat{F}(L, p), \hat{F}(R, p)\}$
  for $p' = 1$; $p' < p$; $p' := p' + 1$ **do**
    $S := \hat{F}(L, p') + \hat{F}(R, p - p')$ ▷ empty solutions for $p' = 0$ in R or L, respectively
    if $S < \hat{F}(X(u), p)$ **then** $\hat{F}(X(u), p) := S$