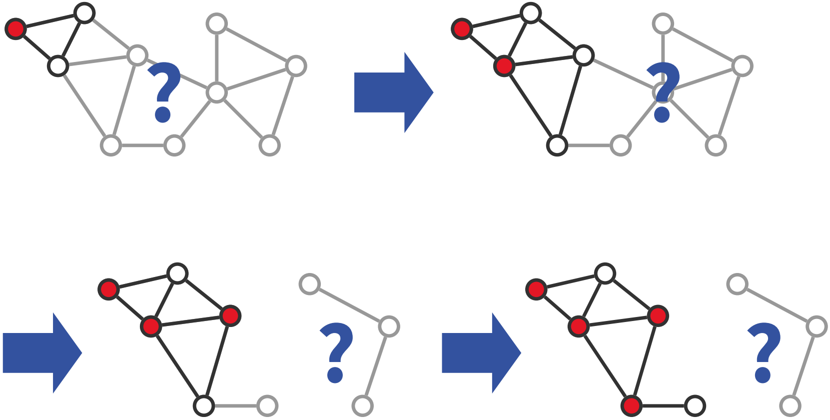
Adaptive algorithm for finding connected dominating sets in uncertain graphs

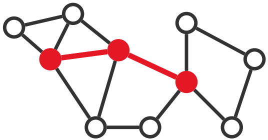
Takuro Fukunaga

Chuo University

Search on an uncertain graph



Connected dominating set



CDS

- undirected graph $G = (V, E)$
- $D \subseteq V$: **connected dominating set (CDS)**

$$\Updownarrow$$

  - $\forall v \in V$ is in D or has a neighbor in D
  - D induces a connected subgraph

CDS Problem

Find a minimum weight CDS.

There are many studies because it has an application in wireless ad hoc networks.

Scenario-based stochastic setting

scenarios $(V_1, p_1), \ldots, (V_k, p_k)$

- $V_i \subseteq V$ is the set of active nodes
- p_i is the probability that V_i is realized
- the starting node s is active in any scenario

Problem

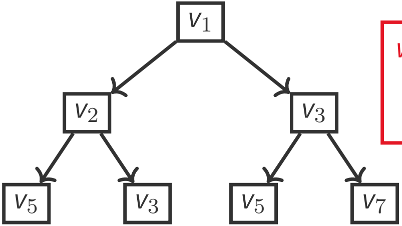
Input:

- undirected graph $G = (V, E)$
- node weights $w\colon V \to \mathbb{R}_+$
- starting node $s \in V$
- scenarios $(V_1, p_1), \ldots, (V_k, p_k)$

Output: A minimum weight CDS including s of $G[A]$, where A is the connected component including s in the realized scenario.

Adaptive policy

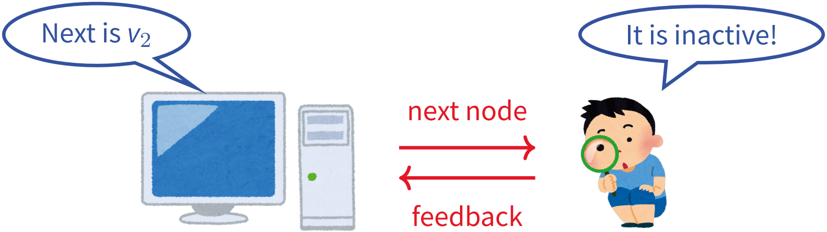
- Nodes are chosen sequentially.
- After a node is chosen, the states (i.e., $\in V_i$ or $\notin V_i$?) of some nodes are revealed as a feedback.
- The subsequent behavior depends on the feedback.
- The performance of policy π is measured by $w(\pi) := \mathbb{E}[w(\text{nodes chosen by } \pi)]$.
- Approximation ratio of $\pi := w(\pi) / \min_{\text{adaptive } \pi'} w(\pi')$,



$$w(\pi) = 5 \times .4 + 3 \times .2 + 4 \times .1 + 9 \times .3 = 5.7$$

weights	5	3	4	9
probability	.4	.2	.1	.3

Output of our algorithm



- Our objective is to design an algorithm that computes an adaptive policy of better approximation ratio.
- Our algorithm does not output the decision tree explicitly. It receives the feedback and outputs the node to be chosen next in polynomial time.

Feedback models

Full feedback model

- When v is chosen, the states of all neighbors of v are revealed.
- We always choose nodes that are revealed to be active.

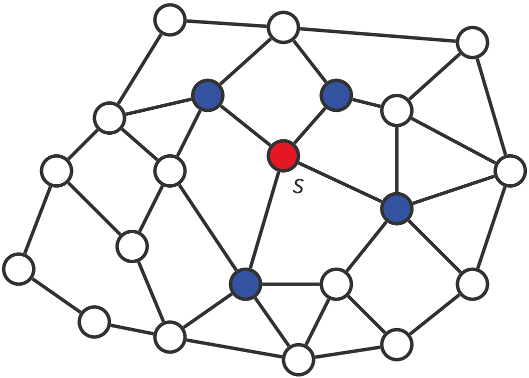
Partial feedback model

- When v is chosen, the state of v is revealed.

In both models,

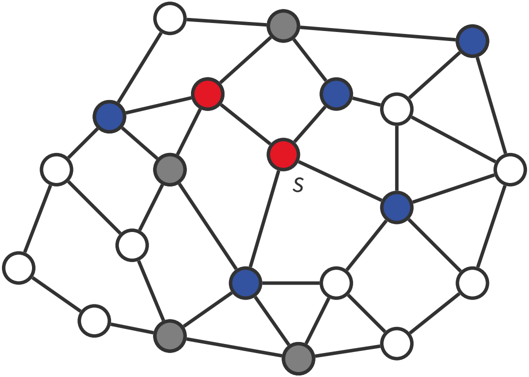
- The objective counts the weights of the chosen nodes (including those that are inactive).
- We have to continue choosing until the current solution is confirmed to be a CDS.

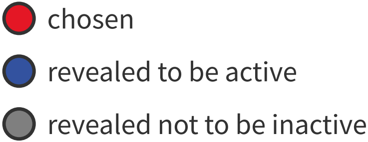
Example: full feedback model



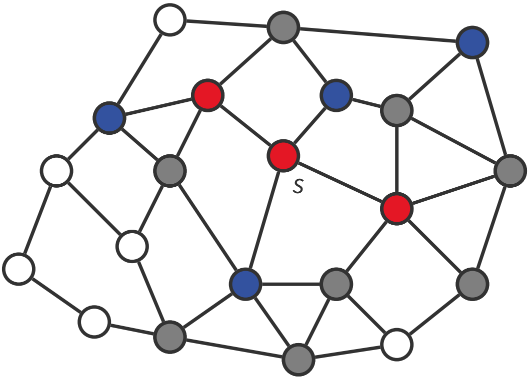
- chosen
- revealed to be active
- revealed not to be inactive

Example: full feedback model





Example: full feedback model

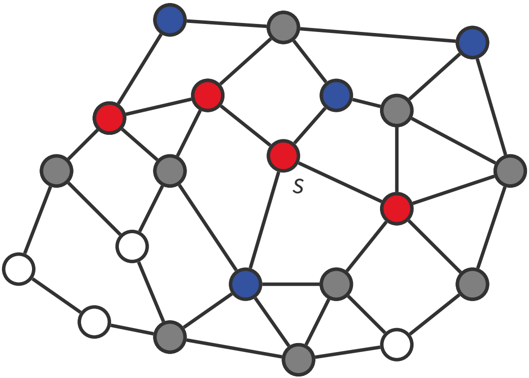


chosen

revealed to be active

revealed not to be inactive

Example: full feedback model



chosen

revealed to be active

revealed not to be inactive

Our contribution

- α: approximation factor for the node-weighted polymatroid Steiner tree
- $m = \max_i \frac{1}{p_i}$

Contribution

$O(\alpha \log m)$-approximation adaptive algorithms both for full feedback and for partial feedback models.

Related studies

- Non-stochastic version of the CDS problem has an $O(\log |V|)$-approximation for general graphs, and a constant-approximation for unit disk graphs.
- Adaptive shorted path problem (Canadian Traveler Problem) has an $O(\alpha \log m)$-approximation [Lim et al. UAI17] (α is for the edge-weighted polymatroid Steiner tree)

Polymatroid Steiner tree

Input

- undirected graph $G = (V, E)$
- edge- or node-weights w
- monotone submodular function $f : 2^V \to \mathbb{R}_+$

A tree T of G is called a polymatroid Steiner tree if $f(V_T) = f(V)$.

Polymatroid Steiner tree problem

Find a minimum weight polymatroid Steiner tree.

- edge-weights: $O(\frac{1}{\epsilon \log \log |V|} \log^{2+\epsilon} |V| \log f(V))$-approximation [Călinescu, Zelikovsky, 05]
- node-weights: for unit disk graphs, it can be reduced to edge-weights within a constant-approximation factor

Idea

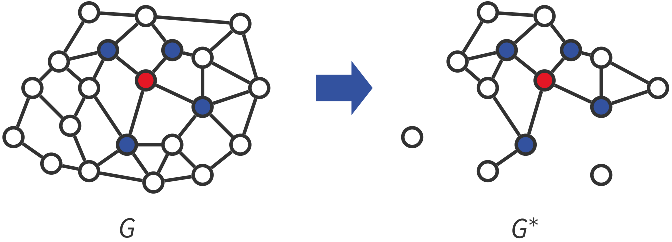
We rely on two ideas used in [Lim et al. UAI17]

- **Idea 1:** We infer the state of each node as the most likely state
- **Idea 2:** On the inferred graph, we compute an **exploring solution** and an **exploiting solution**, and choose nodes from the smaller-weight one.

Computing exploring and exploiting solutions can be formulated as the polymatroid Steiner tree problem.

Outline of our algorithm

1. Define G^* as the graph induced by
$V^* := \{v \in V : \Pr[v \textbf{ active}] \geq 0.5\}$ **(infer the unobserved states)**



2. Computing an exploring solution U_1 and an exploiting solution U_2 in G^*

3. Choose nodes sequentially from $\arg\min\{w(U_1), w(U_2)\}$

- inference of G^* is wrong
  $\Rightarrow$ stop the choice and return to Step 1
- all nodes in U_1 are chosen $\Rightarrow$ return to Step 1
- all nodes in U_2 are chosen $\Rightarrow$ terminate

Exploring solution

Exploring solution

If all nodes in the solution are chosen and the states are observed as inferred, then at least half of the residual scenarios are rejected.

Finding a minimum weight exploring solution is a polymatroid Steiner tree problem.

- $R_U := \{$scenarios rejected by observing the states of U as inferred$\}$ $(\forall U \subseteq V)$
- monotone submodular function $f\colon 2^V \to \mathbb{R}_+$

$$f(U) := \min \left\{ \frac{1}{2}, \sum_{i \in R_U} p_i \right\}$$

- An exploring solution is a node set $U \subseteq V$ such that $f(U) = 1/2$ and it induces a connected subgraph

Exploiting solution

Exploiting solution

If all nodes in the solution are chosen and the states are observed as inferred, then the chosen nodes are confirmed to form a CDS.

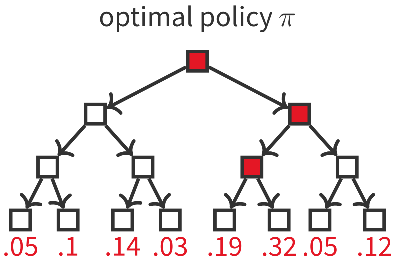
Finding a minimum weight exploiting solution is a polymatroid Steiner tree problem.

- $S :=$ set of unobserved nodes
- $u \not\sim_i v$ means that u and v are not connected in $G[V_i]$
- monotone submodular function $g \colon 2^V \to \mathbb{R}_+$

$$g(U) := |S \cap N(U)| + \sum_{v \in S \setminus N(U)} \sum_{i : v \not\sim_i r} p_i$$

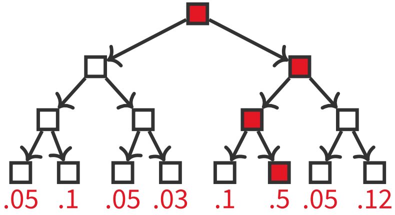
- exploiting solution is a node set $U \subseteq V$ such that $g(U) = g(V)$ and it induces a connected subgraph and

Analysis



Any scenario happens with probability ≤ 0.5

$$w(\text{exploring sol}) \leq 2w(\pi)$$



A scenario happens with probability > 0.5

$$w(\text{exploiting solution}) \leq 2w(\pi)$$

Analysis (cont)

of iterations

- all nodes in the exploring solution are chosen or observing that the inference is wrong ➜ more than a half scenarios are rejected
- choosing all nodes in the exploiting solution ➜ terminate

$$\Rightarrow \text{\# of iterations} = O(\log m)$$

Theorem

If we have an α-approximation algorithm for the polymatroid Steiner tree, for any adaptive policy π, we have

$$\mathbb{E}[\text{weights of output solution}] = O(\alpha \log m) w(\pi).$$

Conclusion

Summary

- A stochastic version of the CDS problem is considered.
- We present an adaptive algorithm for it.
- Key is to formulate computing an exploring and an exploiting solution as the polymatroid Steiner tree problems.

Open problem

- How if states of nodes are independent?
- Other combinatorial optimization problems?