

Lecture 6: September 24, 2001

*Lecturer: Micah Adler**Scribe: Shangzhu Wang*

6.1 Review From Last Lecture

We'll go over what we did in the last lecture with regard to subset systems and greedy algorithms.

6.1.1 Subset Systems

A subset system (E, I) consists of

E : A finite set of elements

I : A collection of independent subsets of E that is closed under inclusion.

Closed under inclusion means that if $i \in I$ and $i' \subseteq i$, then $i' \in I$.

Note: We can think of the independent subsets as "valid" subsets of E that solve the optimization problem that we are considering.

6.1.2 Optimization Problems on Subset Systems

We can define a generic optimization problem for subset systems.

Input: $(E, I), w : \rightarrow \mathbb{R}^+$

Output: the element of I with maximum weight (that is, a valid subset of E with maximum weight)

6.1.3 The Greedy Algorithm for Optimization Problems

The generic greedy algorithm for optimization problems is as follows:

```
 $i = \emptyset$ 
sort the elements of  $E$  by non-decreasing weight
for each  $e \in E$  in turn
    if  $i + e \in I$  then  $i = i + e$ 
output  $i$ 
end for each
```

In order to characterize the subset systems for which the greedy algorithm provides an optimal solution as well as those for which it does not, we use *matroids*.

6.1.4 Matroid

A **matroid** is a subset system which satisfies the *exchange property*.

The **exchange property**:

If $i, i' \in I$ such that $|i| < |i'|$ **then**
 $\exists e \in i' - i$ such that $i + e \in I$

In other words, given two subsets from I , when one of the subsets has a larger number of elements than the other, there will be some element that is in the larger set and not in the smaller which can be added to the smaller to produce another valid subset.

6.1.5 The Cardinality Theorem

Theorem 6.1 *The Cardinality Theorem*

A subset system (E, I) is a matroid if and only if $\forall A \subseteq E$ if i and i' are maximal independent subsets of A then $|i| = |i'|$.

Note that *the Cardinality Theorem* is an alternative definition of a matroid.

6.2 Proof that the Greedy Algorithm Solves the Optimization Problem iff the Subset System is a Matroid

Theorem 6.2 *The Greedy Algorithm solves the optimization problem for (E, I) if and only if (E, I) is a matroid.*

Proof:

Show both directions for any weight function.

(\Leftarrow): If (E, I) is a matroid, then the Greedy Algorithm produces the optimal solution.

Assume that (E, I) is a matroid, but that the Greedy Algorithm does **not** find the optimal solution.

Let the greedy solution be

$$i = \{e_1, e_2, \dots, e_k\}$$

and the optimal solution be

$$j = \{e'_1, e'_2, \dots, e'_k\}$$

It's obvious that $w(j) > w(i)$. And also note that the two sets have the same size. By the definition of the Greedy Algorithm, it will produce the maximal solution. Also the optimal solution must be the maximal solution. Therefore, by *the Cardinality Theorem*, we get $|i| = |j|$.

We assume an ordering of the elements of i

$$w(e_1) \geq w(e_2) \geq \dots \geq w(e_k)$$

and j

$$w(e'_1) \geq w(e'_2) \geq \dots \geq w(e'_k)$$

Let s be the smallest index such that the $w(e'_s) > w(e_s)$.

Let

$$\alpha = \{e_1, e_2, \dots, e_s - 1\}$$

and

$$\beta = \{e'_1, e'_2, \dots, e'_s - 1, e'_s\}$$

Keep in mind that we have a matroid. Then by the *Exchange Property*, $\exists e'_t \in \beta - \alpha$ such that $\alpha \cup \{e'_t\} \in I$. In other words, there exists an element of β that can be added to α to produce a set in I .

It's obvious that $w(e'_t)$ is at least as large as $w(e'_s)$, and we also have $w(e'_s) > w(e_s)$. That is $w(e'_t) \geq w(e'_s) > w(e_s)$. Therefore, $w(e'_t) > w(e_s)$.

However, if i is truly the greedy solution, then the Greedy Algorithm should have considered e'_t before e_s . This is a contradiction.

(\Rightarrow): If the Greedy Algorithm gives the optimal solution, then (E, I) is a matroid.

Assume that (E, I) is **not** a matroid. We will show that, for a suitable weight function, the Greedy Algorithm fails.

Because (E, I) is not a matroid, there exists $i, i' \in I$ such that $|i| < |i'|$ and $\neg \exists e \in i' - i$ such that $i + e \in I$. Let $m = |i|$. Here we present a weight function for which the Greedy Algorithm fails:

$$w(e) = \begin{cases} m + 2 & \text{if } e \in i \\ m + 1 & \text{if } e \in i' - i \\ 0 & \text{otherwise} \end{cases}$$

Note that since weight function is always positive, here "0" is in fact a very very small positive number.

With this weight function, the Greedy Algorithm produces a solution equal to the set i . Therefore, $w(i) = m(m + 2) = m^2 + 2m$.

The Optimal Solution contains all of i' , which contains at least $m + 1$ elements, $w(i) \geq (m + 1)^2 = m^2 + 2m + 1$.

Therefore, $w(i) < w(i')$, so the Greedy Algorithm failed to find an optimal solution. ■

6.2.1 Example 1

Input: A weighted directed graph $G = (V, E)$. $w : E \Rightarrow \mathbb{R}^+$.

Output: The maximal weight subset of E such that no two edges point to the same node.

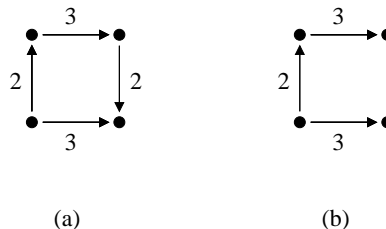


Figure 6.1: (a) Example of a weighted directed graph. (b) Optimal Solution to the graph in (a)

Prove that, in general, the Greedy Algorithm does work on this problem.

E : edges of the graph.
 I : subsets of edges in which no more than one edge points to a vertex.

We are going to show that (E,I) is in fact a matroid.

For any subset $A \in E$, the number of edges in the *maximal* valid subset of A is the number of vertices in A with at least one incoming edge. According to the Cardinality Theorem, this is a matroid, and the Greedy Algorithm will find the optimal solution.

6.2.2 Example 2

Input: $n \times m$ matrix M , with real numbered entries. Weight for each column of M .
Output: Maximum weight set of linearly independent columns.

$$V_n = \begin{pmatrix} 3 & 1 & 3 & 0 & 2 & 1 & 2 \\ 0 & 2 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 2 & 0 & 1 & 0 & 0 \\ 2 & 0 & 0 & -1 & 0 & 2 & 3 \end{pmatrix}$$

Subset System:

E : columns of M .
 I : subsets of columns that are linearly independent.

From the linear algebra, we know that all maximal linearly independent subsets of a set of vectors have the same cardinality. Therefore, this is a matroid.
 Note:It doesn't matter what the weighting function is.

6.3 The Bipartite Matching Problem

Input: A bipartite graph $B = (U, V, E)$. U and V are distinct sets of vertices, and E a set of edges satisfying the property that each edge spans from a vertex in U to a vertex in V .
Output: A matching (no two edges share a common vertex) of maximum size.

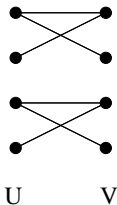


Figure 6.2: Example of a Bipartite Graph

6.3.1 Application for the Bipartite Matching Problem

The Bipartite Matching Problem has a lot of applications. For example, we have a group of people that we would like to assign to a number of tasks. We also have

some restrictions on that:

1. at most one task per person
2. at most one person per task

The goal is to maximize the number of tasks performed.

This is equivalent to the bipartite matching problem. U corresponds to individual people, V to the tasks, and E to what task a person can perform.

6.3.2 Solution to the Bipartite Matching Problem

The Bipartite Matching Problem is not actually a matroid, but we can define the problem *in terms of matroid*.

First, treat U independently of V . According to the restriction on U (at most one edge is incident to any node of U), we have: A matroid (E, I) ; I : at most one edge per node of U .

Then we treat V independently of U . Again, according to the restrictions on V , we have: A matroid (E, I') ; I' : at most one edge per node of V .

Thus, the valid matching is: subsets in $I \cap I'$.

Then, there arises the problem of **intersection of matroids**.

6.3.3 Time bound on maximum cardinality set

Theorem 6.3 Let $M = (E, I)$ and $N = (E, I')$ be two matroids over elements E . We can find the maximum cardinality set in $I \cap I'$ in time $O(|E|^3 * C(E, I, I'))$, where $C(E, I, I')$ is the time required to test membership in I and I' .

6.3.4 Augmenting Path

To find a maximum sized matching, a key technique is the *augmenting path*.

Let M be any bipartite matching,

Definition 6.4 A free vertex is a node not incident to edge in M

Definition 6.5 An augmenting path P for M is a path for alternating matching and non-matching edges starting and ending at free vertex.

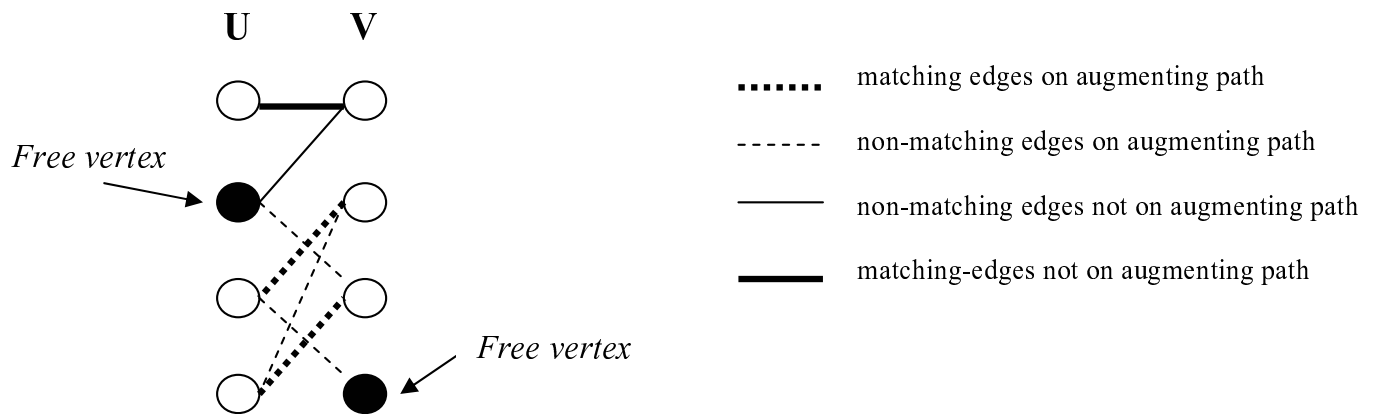


Figure 6.3: Example of an augmenting path

We can use augmenting paths to build matchings one edge at a time, however there may be backtracking necessary since this is not a matroid problem.

References

- 1 KELLY PORPIGLIA, and KAT HANNA, Scribe Notes for Lecture 5, UMASS course 611: Advanced Algorithms, Fall 2000.
- 2 MATTHEW HERTZ, and JOEL SIEH, Scribe Notes for Lecture 6, UMASS course 611: Advanced Algorithms, Fall 2000.