CS 524: Introduction to Optimization
Lecture 9 : Assignment and networks

Michael Ferris

Computer Sciences Department
University of Wisconsin-Madison

September 25, 2023

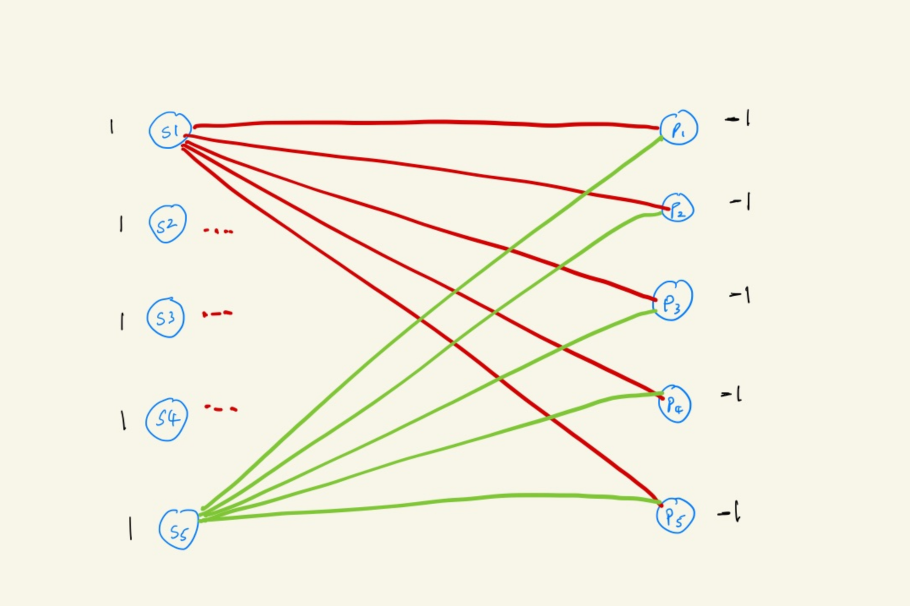
Assignment Problems

The Story

- A teacher wishes to assign each of 5 projects to 5 different students.
- Each student has indicated their preference for each project by assigning it a score between 0 and 10.
- (0 indicating strong dislike and 10 indicating strong preference).
- The teacher wishes to make the assignment of projects to students in a way that maximizes the overall satisfaction
  - as measured by the sum of the preferences for the given assignments.

- How do we model this?

Assignment



Assignment Problem

Variables

- $x_{ij} = \begin{cases} 1 & \text{if student } i \text{ is assigned to project } j \\ 0 & \text{Otherwise} \end{cases}$

Sets

- I: Set of students
- J: Set of Projects

Parameters

- c_{ij}: Preference of student $i \in I$ for project $j \in J$

$$\max \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\begin{aligned} \sum_{i \in I} x_{ij} &= 1 \quad \forall j \in J \\ \sum_{j \in J} x_{ij} &= 1 \quad \forall i \in I \\ x_{ij} &\in \{0,1\} \quad \forall i \in I, j \in J \end{aligned}$$

Assignment Problem

- What we've just modeled is known as an assignment problem
  - We are assigning objects in an optimal way
  - There are lots of applications
- It can be visualized in terms of a graph representation.
- A graph G is a set V of vertices (nodes) and a set $E \subseteq V \times V$ of edges
  - They are drawn with "Points and lines"
  - The graph is "undirected", meaning the edges do not have arrows!

Some Definitions

- A graph $G = (V, E)$ is bipartite if V can be partitioned into two sets $V = L \cup R$ with edges only between the two sets: $e = (i, j) \in E \Rightarrow ((i \in L) \cap (j \in R)) \cup ((i \in R) \cap (j \in L))$
- A matching is a subset of edges $M \subseteq E$ such that for all $v \in V$, ≤ 1 edge of M is incident upon it.
- A perfect matching is a subset of edges $M \subseteq E$ such that for all $v \in V$, exactly 1 edge of M is incident upon it.

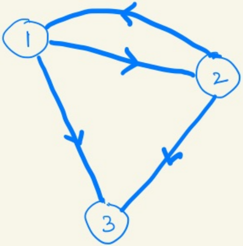
Weighted Matching

- The assignment problem is to find a maximum-weight perfect matching in a bipartite graph
- There are faster algorithms than the simplex method for this problem. The Hungarian Method: $O(n^3)$, other algorithms have complexity $O(m + n^2)$, m is number of edges
- There is an algorithm called the "network simplex method" which takes into account that the matrix A is a network matrix.

Network

- A network $G = (N, A)$ consists of a set of nodes N and a set of arcs $A \subseteq N \times N$
- Usually people define a network as a directed graph. The arcs $a = (i, j)$ are ordered:
  - They have a head (to) and a tail (from)
  - Drawn with arrows
- Note: Any undirected network can be modeled as a directed network with two arcs: one directed in each direction.
- In a capacitated network the arcs $a \in A$ have capacities u_a

Simple network



$$
\begin{array}{c}
\\ \textcircled{1} \\ \textcircled{2} \\ \textcircled{3}
\end{array}
\begin{array}{c}
\begin{array}{cccc} (1,2) & (2,1) & (1,3) & (2,3) \end{array} \\
\begin{bmatrix} 1 & -1 & 1 & \\ -1 & 1 & & 1 \\ & & -1 & -1 \end{bmatrix}
\end{array}
$$

tail $\longrightarrow$ head

Node-Arc Incidence Matrices

- Row for every node i
- Column for every arc $k = (i, j)$
- $a_{ik} = 1$ if arc k leaves node i
- $a_{ik} =$ -1 if arc k enters node i

TU

Totally Unimodular

Matrix A is totally unimodular (TU) if every subdeterminant (determinant of square submatrix) of A has value $+1$, -1, or 0.

TU = Integer Extreme Points

Let $X = \{x \in \mathbb{R}^n_+ \mid Ax \leq b\}$.

- A is TU if and only if all extreme points of X are integer valued for any integer vector $b \in \mathbb{Z}^m$

Cool Facts

- Network (node-arc incidence) matrices are *TU*
- Node(Vertex)-edge incidence matrices of bipartite graphs are TU
- Node-edge incidence matrices of non-bipartite graphs are not TU

Something for nothing

- If we solve a linear program over X whose constraint matrix A is TU and b is integer valued using the simplex method (that produces an extreme point solution) then that solution will be integer valued
- No need to impose integrality (binary, integer) on variables

TU matrices

- If A is TU, so it A^T
- If A is TU then $\begin{bmatrix} A \\ I \end{bmatrix}$ is TU
- If A is TU then $\begin{bmatrix} A \\ -A \end{bmatrix}$ is TU

Assignment Problem (assignprefs.gms)

Variables

- $x_{ij} = \begin{cases} 1 & \text{if student } i \text{ is assigned to project } j \\ 0 & \text{Otherwise} \end{cases}$

Sets

- I: Set of students
- J: Set of Projects

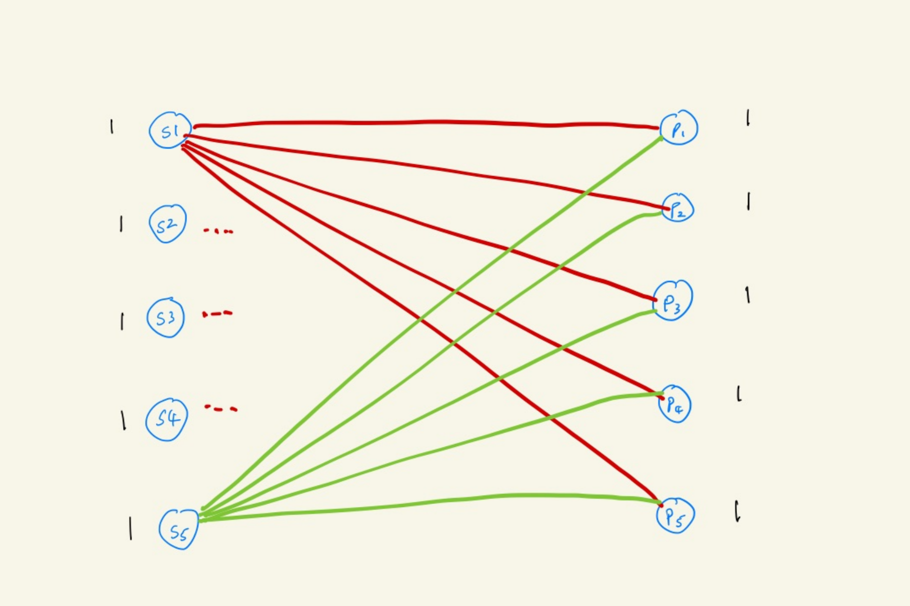
Parameters

- c_{ij}: Preference of student $i \in I$ for project $j \in J$

$$\max \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\begin{aligned} \sum_{i \in I} x_{ij} &= 1 \quad \forall j \in J \\ \sum_{j \in J} x_{ij} &= 1 \quad \forall i \in I \\ x_{ij} &\in \{0,1\} \quad \forall i \in I, j \in J \end{aligned}$$

Assignment



Exercise: `assignprefs.gms`

- Node-arc incidence matrix has 10 rows and 25 columns
- Using model types: `mip` and `rmip` with `binary variables`
- Constraints form TU matrix with integer right hand side
- Automatically detect network using cplex options

Setting Options

Option Settings for GAMS/CPLEX.

- To change LP solver, use `option lp=cplex;`
- Each solver (like CPLEX) has lots of options that might help improve solver performance. See documentation!
- Options are mostly for advanced users, but can improve performance, especially for integer programs.
- To actually set options include this command in the file, after the "model" statement but before the "solve": `modelname.optfile=1;`
- Then build a file in the same directory as the GAMS file, called "cplex.opt" (in the case of the cplex solver) which lists the option names followed by the chosen value.

Example Option Setting

- One can specialise CPLEX for network problems. For example for a network problem, `cplex.opt` could contain

```
lpmethod 3
netfind 2
preind 0
```

- Can generate the cplex.opt file from within the gams file by

```
$onecho > cplex.opt
lpmethod 3
netfind 2
preind 0
$offecho
```