Combinatorial Optimisation, MSc in Applied Statistics, Hilary Term 2010. Notes to accompany lectures, chapters 6 and 7. Earlier notes covered chapters 1 – 5.

6 Matchings

6.1 Introduction

A *matching* in a graph G is a subset M of the edges such that no two edges in M share the same node. insert figure

A node of G is *covered* if some edge in M is incident with it and otherwise is *exposed*. A matching is *maximum* if it is of maximal size. It is *perfect* (or complete) if no nodes are exposed. A path (without repeated vertices) is M-*alternating* if the edges are alternately in M and out of M eg 2,4,5,6 above. An M-alternating path is M-*augmenting* if both end vertices are exposed eg 1,2,3,5,4,6. Observe that an M-augmenting path has one more edge not in M than in M. We may identify a path with its set of edges.

Lemma 6.1 *Let M be a matching and P an M-augmenting path. Then $M' = M \triangle P$ is a matching of cardinality $|M| + 1$.*

Here the *symmetric difference* $A \triangle B$ of two sets A and B is the set of elements in exactly one of the sets: thus $A \triangle B$ may also be written as $(A \setminus B) \cup (B \setminus A)$. To form M' we may start with M, and then in the path P we swap which edges go into M'.

Proof Clearly $|M'| = |M| + 1$. Let e, e' be distinct edges in M'. We must check that they can have no node in common. There are three cases: (i) $e, e' \in M \setminus P$, trivial, since M is a matching; (ii) $e, e' \in P \setminus M$, ok since e and e' are not successive on P, and P has no repeated vertices; and (iii) $e \in M \setminus P, e' \in P \setminus M$, ok since e cannot have an end node in P (not an internal node v of P, since a matching edge in P is incident with v; and not an end node of P, since both end nodes are exposed). □

Theorem 6.2 *(Berge 1957) A matching M in a graph G is maximum if and only if there is no M-augmenting path.*

Proof (⇒) done by Lemma 6.1.
(⇐) Suppose M is not maximum. Let M' be a matching with $|M'| > |M|$. Consider the edges in $M \triangle M'$. In the corresponding subgraph H of G each vertex can be incident with at most one edge of M and one of M'. Hence each component of H is either an even cycle or a path, with edges alternately in M and M'. But H has more M' edges than M edges. Thus some path component of H has more M' edges than M edges, and it then forms an M-augmenting path. □

So to find a maximum matching we can start with say $M = \phi$ and repeatedly search for augmenting paths. But how? Before we find out how to go let us note a way to stop. A *cover* in G is a set K of nodes such that every edge of G has at least one end in K.

Lemma 6.3 *Let M be a matching and K a cover in G. Then $|M| \leq |K|$.*

Proof Each edge in M must contain at least one node in K. □

insert figure of C_5?

If above we have $|M| = |K|$ then of course M is a maximum matching (and K is a minimum cover, and K consists of exactly one end node of each edge in M). Note that in the 5-cycle C_5 we cannot have equality. However, for bipartite graphs (see below) we shall have equality and we can base an algorithm for maximum matchings on Lemmas 6.1 and 6.3.
A graph $G = (V, E)$ is *bipartite* if its nodes may be partitioned as $V = S \cup T$ such that each edge $e \in E$ has one end node in S and one in T. It may be shown that a graph is bipartite if and only if it contains no odd cycle. Further, given a graph G on n nodes, in $O(n^2)$ time we may test if it is bipartite and if so output a corresponding partition $S \cup T$. We shall often use the notation $G = (S, T, E)$.

6.2 Bipartite maximum cardinality matching algorithm

The 'Hungarian algorithm' may be described as follows. The input is a bipartite graph $G = (S, T, E)$.

Start

Start with a matching M in G, perhaps empty.
No nodes are labelled or scanned.

Tree building

give the label 0 to each exposed node in S.

while there is no breakthrough and there is a labelled unscanned node
find a labelled unscanned node i

if the node $i \in S$ **then** scan it as follows: for each edge $\{i, j\} \notin M$ incident to node i, give node j the label 'i' if node j is not already labelled, and declare 'breakthrough' if node j is exposed

if the node $i \in T$ **then** scan it as follows: find the unique edge $\{i, j\} \in M$ incident to node i, and give node j the label 'i'.

Augment

if there has just been a breakthrough **then**
start at the breakthrough node j and backtrack using the labels
to find an M-augmenting path, augment M, remove all labels,
and return to the step *Tree building*.

Hungarian labelling

return the matching M and the set K of the unlabelled nodes in S and the labelled nodes in T.

example

Theorem 6.4 *The algorithm returns a matching M and a cover K with $|M| = |K|$ in $O(m^2n)$ steps, where $|S| = m$ and $|T| = n$, and we assume that $m \leq n$.*

Proof Let us say that we enter a new *stage* each time we enter the 'tree-building' step. During each stage, we scan nodes $i \in S$ at most m times, and each performance takes $O(n)$ steps. This then takes $O(mn)$ steps. Considering scanning a node $i \in T$ similarly, we see that during each stage scanning takes $O(mn)$ steps. Also, by lemma 6.1 the augmentation step correctly makes the matching one larger, within time $O(mn)$.

Now there are $O(m)$ stages, each takes $O(mn)$ steps, and throughout we have a proper matching. Hence after $O(m^2n)$ steps we must stop, with a matching M and the set K.

We **claim** first that K is a cover (of edges by nodes). To establish this we must show that there can be no edge between a labelled node $i \in S$ and an unlabelled node $j \in T$. But there can be no such edge in M, since then i would have received its label from j; and there can be no such edge out of M since then i would give a label to j.

We **claim** further that K consists of exactly one node from each edge in M, and so $|M| = |K|$. To see this, note first that K contains only covered nodes, since all exposed nodes in S are labelled and we have not labelled any exposed node in T. Also, for each edge ij in M where $i \in S$ and $j \in T$, either both or neither of its end nodes i and j are labelled, since if j is labelled then it gives a label to i and if j is not labelled then i cannot be labelled. If both i and j are labelled then j goes in K, and if neither is labelled then i goes in K. □

6.3 König's theorem and Hall's theorem

Let G be a bipartite graph. We observed that $|M| \leq |K|$ for any matching M and cover K. We described an algorithm which yields a matching M and cover K with $|M| = |K|$. We have thus also proved

Theorem 6.5 *(König's theorem) In a bipartite graph,* $\max |M| = \min |K|$.

This result was originally phrased as follows. In a 0–1 matrix the maximum number of 1's no two on the same line (that is, row or column) equals the minimum number of lines to cover all 1's. It is easy to deduce Hall's 'marriage theorem' from König's theorem (or vice versa). We suppose that some pairs of boys and girls know each other.

Theorem 6.6 *(Hall's theorem) We can marry all the boys to girls they know if and only if for each k, each set of k boys between them know at least k girls.*

Proof The condition is clearly necessary. Suppose now that it holds. Form the bipartite graph $H = (B, G, E)$ where B, G are the sets of boys, girls respectively, and $\{b, g\}$ is an edge if boy b and girl g know each other. We must show that there is a matching in H of size $|B|$.

Let K be a cover. By König's theorem it suffices to show that $|K| \geq |B|$.

insert figure

All girls known to boys in $B \backslash K$ must be in $G \cap K$. Thus by the Hall condition, we have $|B \backslash K| \leq |G \cap K|$. Hence

$$|K| = |B \cap K| + |G \cap K| \geq |B \cap K| + |B \backslash K| = |B|,$$

and so $|K| \geq |B|$, as required. □

7 Assignment Problem

7.1 Introduction

Suppose that we are to assign n jobs to n machines, one job per machine (or say n pilots to n planes). Job i assigned to machine j incurs cost $c_{ij} \geq 0$, and we seek an assignment with least total cost. In other words, we seek a minimum cost perfect matching in the corresponding complete bipartite graph, with nodes $R_1, \ldots, R_n$ and $C_1, \ldots, C_n$ and for each i, j an edge $R_i C_j$ joining R_i and C_j with cost c_{ij}.

Our algorithm for the assignment problem is best understood in the framework of LP (linear programming) duality. However, let us ignore LP for now and return to that later.

Call real numbers $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{v} = (v_1, \ldots, v_n)$ *dual-feasible* if the *reduced cost* $\bar{c}_{ij} = c_{ij} - u_i - v_j$ is ≥ 0 for each $i, j = 1, \ldots, n$. Given dual feasible $\mathbf{u}, \mathbf{v}$ the *equality graph* $G_{\mathbf{u},\mathbf{v}}$ is the bipartite graph with nodes $R_1, \ldots, R_n$ and $C_1, \ldots, C_n$ and an edge joining R_i and C_j whenever the reduced cost $\bar{c}_{ij} = 0$.

Lemma 7.1 *Let $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{v} = (v_1, \ldots, v_n)$ be dual-feasible, and let M^* be a perfect matching in the equality graph $G_{\mathbf{u},\mathbf{v}}$. Then M^* is an optimal assignment.*

Proof For any perfect matching M in the complete bipartite graph,

$$\text{cost of } M = \sum_{R_i C_j \in M} c_{ij} \geq \sum_{R_i C_j \in M} (u_i + v_j) = \sum_i u_i + \sum_j v_j$$

with equality for M^*, so that

$$\text{cost of } M \geq \sum_i u_i + \sum_j v_j = \text{cost of } M^*.$$

□

7.2 Hungarian method for the assignment problem

Our approach to solving the assignment problem is to maintain throughout a dual feasible solution and a matching in the corresponding equality graph, and to seek a perfect matching. The algorithm may be described as follows.

The input is an $n \times n$ matrix of costs $c_{ij} \geq 0$. Our main tool is the bipartite maximum cardinality matching algorithm, with the step 'Hungarian labelling' replaced by a new step 'Update dual solution'.

Hungarian method

find dual feasible u_i, v_j (perhaps all 0), construct the equality graph $G_{\mathbf{u},\mathbf{v}}$ and find a matching M in $G_{\mathbf{u},\mathbf{v}}$ (perhaps the empty matching).

while M is not perfect
    do tree-building with M in equality graph
    while not breakthrough
        update dual solution
        continue tree building
    augment M
return M [an optimal matching] and u_i, v_j [which are dual optimal].

The step *update dual solution* is as follows.

Compute $\delta = \min\{\bar{c}_{ij} : i \in S \cap L,\ j \in T \cap \bar{L}\}$, where $\bar{c}_{ij}$ is the reduced cost $c_{ij} - u_i - v_j$, L is the set of labelled vertices, and $\bar{L}$ is the rest. For each $i \in S \cap L$ increase u_i by δ; and for each $j \in T \cap L$ decrease v_j by δ. Construct the new equality graph $G_{\mathbf{u},\mathbf{v}}$.

Example

		$v_1 = 0$	$v_2 = 0$	$v_3 = 0$	$v_4 = 1$
	$u_1 = 2$	5	2	3	4
(c_{ij})	$u_2 = 4$	7	8	4	5
	$u_3 = 3$	6	3	5	6
	$u_4 = 2$	2	2	3	5

$$(\bar{c}_{ij}) \qquad \begin{pmatrix} 3 & 0 & 1 & 1 \\ 3 & 4 & 0 & 0 \\ 3 & 0 & 2 & 2 \\ 0 & 0 & 1 & 2 \end{pmatrix}$$

Observe that $K = (S \cap \bar{L}) \cup (T \cap L) = \{R_2, R_4, C_2\}$ is a cover. Also,

$$\begin{aligned} \delta &= \min\{\bar{c}_{ij} : i \in S \cap L,\ j \in T \cap \bar{L}\} \\ &= \min\{\bar{c}_{ij} \text{ over uncovered elements of matrix}\} \ = \ 1. \end{aligned}$$

Increase u_1 and u_3 by δ, and decrease v_2 by δ.

	0	−1	0	1
3	5	**2**	**3**	**4**
4	7	8	**4**	**5**
4	6	**3**	5	6
2	**2**	2	3	5

The 7 boldface entries above correspond to the edges in the new equality graph $G_{\mathbf{u},\mathbf{v}}$. Observe that each old matching edge and each old tree edge is still in $G_{\mathbf{u},\mathbf{v}}$.

We now obtain the optimal assignment $R_1, C_4; R_2, C_3; R_3, C_2; R_4, C_1$ with reduced cost 0. The original cost is $2+3+4+4 = 13$. Note that we have also an optimal dual solution ($\sum u_i + \sum v_j = 3+4+4+2+0+(-1)+0+1 = 13$).

Theorem 7.2 *The Hungarian method yields an optimal assignment M in $O(n^4)$ steps (together with dual feasible u_i, v_j which verify that M is an optimal solution to (P)).*

Proof Let us say that we are *happy* at some point if we have dual feasible $\mathbf{u}, \mathbf{v}$ and a matching M in the equality graph $G_{\mathbf{u},\mathbf{v}}$. Certainly we start happy. We want to show that we stay happy throughout, and from our earlier work we need only check that the dual update step keeps us happy.

Suppose then that we enter this step happy. Note that M is not perfect, and $\delta > 0$ (since the set K is a cover, that is since there are no edges in the equality graph between $S \cap L$ and $T \cap \bar{L}$). Our changes to the u_i and v_j

change $\bar{c}_{ij} = c_{ij} - u_i - v_j$ as follows:

$$\begin{array}{lrl} \text{decrease } \bar{c}_{ij} \text{ by } \delta & \text{if} & i \in S \cap L,\ j \in T \cap \bar{L} \\ \text{fix } \bar{c}_{ij} & \text{if} & i \in S \cap L,\ j \in T \cap L \\ & \text{or if} & i \in S \cap \bar{L},\ j \in T \cap \bar{L} \\ \text{increase } \bar{c}_{ij} \text{ by } \delta & \text{if} & i \in S \cap \bar{L},\ j \in T \cap L. \end{array}$$

Hence still each $\bar{c}_{ij} \geq 0$. Also, for each edge $\{i, j\}$ in M either each end is labelled or each end is unlabelled, and so $\bar{c}_{ij}$ stays fixed at 0. Hence we shall leave the dual update step happy, as required. Further each old tree edge (along which the last labelling took place) is still in the new equality graph, and we shall be able to label at least one new node in T.

We now know that we stay happy throughout. Also between augmentations we can go round the loop including the dual update step at most n times. Further each loop takes $O(n^2)$ steps. It follows that between augmentations we take $O(n^3)$ steps. But there can be at most n augmentations, and so after $O(n^4)$ steps we must stop. Now we are still happy but we have a perfect matching, and Lemma 7.1 completes the proof. □

With more care we can organise matters so that between augmentations we take $O(n^2)$ steps, and we thus obtain an implementation taking $O(n^3)$ steps (see for example *Network Flows* by Ahuja, Magnanti and Orlin.

7.3 Aside on LP duality

Now let us return to set the method above explicitly in the framework of LP duality. Consider the LP

$$\min \quad \sum_{i,j} c_{ij}\, x_{ij}$$

subject to

$$(P) \qquad \begin{array}{rl} \sum_j x_{ij} = 1 & (i = 1, \ldots, n) \\ \sum_i x_{ij} = 1 & (j = 1, \ldots, n) \\ x_{ij} \geq 0 & (i, j = 1, \ldots, n). \end{array}$$

The integral feasible solutions are the possible assignments or the perfect matchings in the corresponding bipartite graph with costs on the edges. [The

LP is highly degenerate, which can lead to awkwardness with simplex-type methods.]

Now suppose more generally that we are given vectors $\mathbf{c} \in \mathbf{R}^n$ and $\mathbf{b} \in \mathbf{R}^m$ and an $m \times n$ real matrix A. Consider a primal linear programme (P) in the following convenient form: choose $\mathbf{x} \in \mathbf{R}^n$ in order to

$$\begin{array}{lrl} (P) & \min & \mathbf{c}'\mathbf{x} \\ & \text{subject to} & A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}. \end{array}$$

The associated dual programme (D) is then to choose $\mathbf{y} \in \mathbf{R}^m$ in order to

$$\begin{array}{lrl} (D) & \max & \mathbf{y}'\mathbf{b} \\ & \text{subject to} & \mathbf{y}'A \leq \mathbf{c}'. \end{array}$$

The fundamental duality theorem then states that if both (P) and (D) have feasible solutions then both have optimal solutions $\mathbf{x}^*, \mathbf{y}^*$ and the two values are equal (i.e. $\mathbf{c}'\mathbf{x}^* = (\mathbf{y}^*)'\mathbf{b}$). From this follows the

Theorem 7.3 *(Complementary Slackness Theorem)*

Let $\mathbf{x}$ be feasible for (P) and $\mathbf{y}$ for (D). Then both these feasible solutions are optimal if and only if

$$(x_j > 0 \Rightarrow (\mathbf{y}'A)_j = \mathbf{c}_j \qquad \textit{for each } j = 1, \ldots, n).$$

To form the dual of the assignment LP introduce dual variables $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$. Observe that the column of the A matrix corresponding to the primal variable x_{ij} is $\binom{\mathbf{e}_i}{\mathbf{e}_j}$ where $\mathbf{e}_k$ denotes the kth unit vector in $\mathbf{R}^n$. Thus we see that the dual is

$$\begin{array}{lrll} (D) & \max & \sum_i u_i + \sum_j v_j & \\ & \text{subject to} & u_i + v_j \leq c_{ij} & \text{for each } i, j = 1, \ldots, n. \end{array}$$

This explains the earlier of use of the words 'dual-feasible'. The complementary slackness theorem now gives the following extension of Lemma 7.1.

Lemma 7.4 *Let x_{ij} be feasible for (P) and u_i, v_j for (D). Then both are optimal if and only if*

$$(x_{ij} > 0 \Rightarrow u_i + v_j = c_{ij} \qquad \textit{for each } i, j = 1, \ldots, n).$$

We may see that our approach to solving the assignment problem is to maintain throughout both **dual** feasibility and the complementary slackness (CS) conditions, and to seek primal feasibility. (In the simplex method and the transportation algorithm, we maintain primal feasibility and CS and seek dual feasibility.) We end up with a primal optimal solution which is integral, which is thus an optimal assignment.