Discrete Optimization

Ngày 8 tháng 9 năm 2011

Lecture 3: The Assignment Problem

In this lecture we shall learn how to solve the assignment problem efficiently.

Lecture 3: The Assignment Problem

In this lecture we shall learn how to solve the assignment problem efficiently.

(The Hungarian Method)

Observations:

In this lecture we shall learn how to solve the assignment problem efficiently.

(The Hungarian Method)

Observations:

- *Recall: a solution to an $n \times n$ assignment problem is an $n - permutation: \; [i_1, i_2, \ldots i_n]$*

Lecture 3: The Assignment Problem

In this lecture we shall learn how to solve the assignment problem efficiently.

(The Hungarian Method)

Observations:

- *Recall: a solution to an* $n \times n$ *assignment problem is an* $n - \textit{permutation}: \ [i_1, i_2, \dots i_n]$

Lecture 3: The Assignment Problem

In this lecture we shall learn how to solve the assignment problem efficiently.

(The Hungarian Method)

Observations:

- *Recall: a solution to an* $n \times n$ *assignment problem is an* $n-permutation:\ [i_1, i_2, \ldots i_n]$

Definition

Two assignment problems of size n are *equivalent* if any $n - permutation$ which is optimal to one problem is also optimal for the other.

The Hungarian Method, preliminaries

Example

-

	C1	C2	C3	C4				P_1	P_2	P_3	P_4
J1	45	112	114	216			W_1	31	351	123	103
J2	95	52	104	235			W_2	91	51	123	103
J3	90	95	80	180			W_3	81	351	43	103
J4	95	133	141	75			W_4	99	351	123	103

The Hungarian Method, preliminaries

Example

	$C1$	$C2$	$C3$	$C4$				P_1	P_2	P_3	P_4
$J1$	45	112	114	216			W_1	31	351	123	103
$J2$	95	52	104	235			W_2	91	51	123	103
$J3$	90	95	80	180			W_3	81	351	43	103
$J4$	95	133	141	75			W_4	99	351	123	103

The permutation $[1, 2, 3, 4]$ *is clearly the only permutation that gives an optimal solution to both problems.*

The Hungarian Method, preliminaries

Example

	C1	C2	C3	C4				P_1	P_2	P_3	P_4
J1	45	112	114	216			W_1	31	351	123	103
J2	95	52	104	235			W_2	91	51	123	103
J3	90	95	80	180			W_3	81	351	43	103
J4	95	133	141	75			W_4	99	351	123	103

The permutation $[1, 2, 3, 4]$ *is clearly the only permutation that gives an optimal solution to both problems.*

Question

Can you construct an example of a pair of equivalent assignment problems that have five optimal solutions?

(Observations)

(Observations)

- *If we reduce the cost of every bid for job number k by the same amount the new assignment problem will be equivalent to the original problem.*

(Observations)

- *If we reduce the cost of every bid for job number k by the same amount the new assignment problem will be equivalent to the original problem.*
- *If all bids by company number j will be reduced by the same amount the new assignment problem will be equivalent to the original problem.*

(Observations)

- *If we reduce the cost of every bid for job number k by the same amount the new assignment problem will be equivalent to the original problem.*
- *If all bids by company number j will be reduced by the same amount the new assignment problem will be equivalent to the original problem.*
- *If in an assignment problem all entries are non-negative and if there is an assignment whose cost is 0 then it is an optimal assignment.*

The Hungarian Method

The Hungarian method constructs a sequence of equivalent assignment problems until it identifies a 0 *cost assignment.*

The Hungarian Method

The Hungarian method constructs a sequence of equivalent assignment problems until it identifies a 0 *cost assignment.*

The Hungarian Method

The Hungarian method constructs a sequence of equivalent assignment problems until it identifies a 0 *cost assignment.*

This raises the following questions:

The Hungarian Method

The Hungarian method constructs a sequence of equivalent assignment problems until it identifies a 0 *cost assignment.*

This raises the following questions:

Question

Does such an equivalent assignment problem always exist?

The Hungarian Method

The Hungarian method constructs a sequence of equivalent assignment problems until it identifies a 0 *cost assignment.*

This raises the following questions:

Question

Does such an equivalent assignment problem always exist?

The Hungarian Method

The Hungarian method constructs a sequence of equivalent assignment problems until it identifies a 0 cost assignment.

This raises the following questions:

Question

Does such an equivalent assignment problem always exist?

How do we identify a 0 cost assignment?

THE HUNGARIAN METHOD

Definition

A **line** *in an assignment problem is a column or a row.*

THE HUNGARIAN METHOD

Definition

A **line** *in an assignment problem is a column or a row.*

Definition

Two 0 *entries in an assignment problem are* **independent** *if they are not on the same line.*

Definition

A **line** *in an assignment problem is a column or a row.*

Definition

Two 0 *entries in an assignment problem are* **independent** *if they are not on the same line.*

(The Algorithm: reductions)

THE HUNGARIAN METHOD

Definition

A **line** *in an assignment problem is a column or a row.*

Definition

Two 0 *entries in an assignment problem are* **independent** *if they are not on the same line.*

(The Algorithm: reductions)

- *1. Reduce every row by the smallest amount in the row.*

THE HUNGARIAN METHOD

Definition

A **line** *in an assignment problem is a column or a row.*

Definition

Two 0 *entries in an assignment problem are* **independent** *if they are not on the same line.*

(The Algorithm: reductions)

- *1. Reduce every row by the smallest amount in the row.*
- *2. Reduce every column by the smallest amount in the column.*

THE HUNGARIAN METHOD

Definition

A **line** *in an assignment problem is a column or a row.*

Definition

Two 0 *entries in an assignment problem are* **independent** *if they are not on the same line.*

(The Algorithm: reductions)

- *1. Reduce every row by the smallest amount in the row.*
- *2. Reduce every column by the smallest amount in the column.*
- *3. Find the maximum number of independent zeros. If it is n, stop. You found an optimal solution, if not get a new equivalent assignment problem and try to find a bigger independent set of zeros.*

The Major Steps

Throughout this discussion n will be the number of companies and m the size of the current independent set of zeros.

(Finding a maximal set of zeros)

The Major Steps

Throughout this discussion n will be the number of companies and m the size of the current independent set of zeros.

(Finding a maximal set of zeros)

1. A_1 : *Start by a simple greedy selection of zeros. In each row select the first independent zero.*

The Major Steps

Throughout this discussion n will be the number of companies and m the size of the current independent set of zeros.

(Finding a maximal set of zeros)

1. A_1 : *Start by a simple greedy selection of zeros. In each row select the first independent zero.*
2. A_2 : *If you found m independent zeros and* $m < n$*, then either augment it or find m lines that cover all zeros. See details in a coming slide.*

The Major Steps

Throughout this discussion *n* will be the number of companies and *m* the size of the current independent set of zeros.

(Finding a maximal set of zeros)

1. A_1 : *Start by a simple greedy selection of zeros. In each row select the first independent zero.*
2. A_2 : *If you found m independent zeros and $m < n$, then either augment it or find m lines that cover all zeros. See details in a coming slide.*
3. A_3 : *Find the smallest entry d in the current assignment problem which is not covered by the m lines. Note that $d > 0$.*

The Major Steps

Throughout this discussion n will be the number of companies and m the size of the current independent set of zeros.

(Finding a maximal set of zeros)

1. A_1 : *Start by a simple greedy selection of zeros. In each row select the first independent zero.*
2. A_2 : *If you found m independent zeros and $m < n$, then either augment it or find m lines that cover all zeros. See details in a coming slide.*
3. A_3 : *Find the smallest entry d in the current assignment problem which is not covered by the m lines. Note that $d > 0$.*
4. A_4 : *Reduce all entries in the current assignment problem by d.*

The Major Steps

Throughout this discussion n will be the number of companies and m the size of the current independent set of zeros.

(Finding a maximal set of zeros)

1. A_1 : *Start by a simple greedy selection of zeros. In each row select the first independent zero.*
2. A_2 : *If you found m independent zeros and $m < n$, then either augment it or find m lines that cover all zeros. See details in a coming slide.*
3. A_3 : *Find the smallest entry d in the current assignment problem which is not covered by the m lines. Note that $d > 0$.*
4. A_4 : *Reduce all entries in the current assignment problem by d.*
5. A_5 : *Add d to every entry on the m lines.*

The Major Steps

Throughout this discussion n will be the number of companies and m the size of the current independent set of zeros.

(Finding a maximal set of zeros)

1. A_1 : *Start by a simple greedy selection of zeros. In each row select the first independent zero.*
2. A_2 : *If you found m independent zeros and $m < n$, then either augment it or find m lines that cover all zeros. See details in a coming slide.*
3. A_3 : *Find the smallest entry d in the current assignment problem which is not covered by the m lines. Note that $d > 0$.*
4. A_4 : *Reduce all entries in the current assignment problem by d.*
5. A_5 : *Add d to every entry on the m lines.*
6. A_6 : *Go back to step A_1.*

Questions

Question

Questions

Question

1. *Clearly, the algorithm introduces new zeros, but also removes some (the ones on the intersection of two lines). Can this algorithm cycle forever?*

Questions

Question

1. *Clearly, the algorithm introduces new zeros, but also removes some (the ones on the intersection of two lines). Can this algorithm cycle forever?*
2. *Can you always find m lines that cover all zeros?*

Questions

Question

1. *Clearly, the algorithm introduces new zeros, but also removes some (the ones on the intersection of two lines). Can this algorithm cycle forever?*
2. *Can you always find m lines that cover all zeros?*
3. *How can you find the m lines?*

Questions

Question

1. *Clearly, the algorithm introduces new zeros, but also removes some (the ones on the intersection of two lines). Can this algorithm cycle forever?*
2. *Can you always find m lines that cover all zeros?*
3. *How can you find the m lines?*

Answer

Claim: the algorithm never returns a previous assignment instance.

Questions

Question

1. *Clearly, the algorithm introduces new zeros, but also removes some (the ones on the intersection of two lines). Can this algorithm cycle forever?*
2. *Can you always find m lines that cover all zeros?*
3. *How can you find the m lines?*

Answer

Claim: the algorithm never returns a previous assignment instance.

Questions

Question

1. *Clearly, the algorithm introduces new zeros, but also removes some (the ones on the intersection of two lines). Can this algorithm cycle forever?*
2. *Can you always find m lines that cover all zeros?*
3. *How can you find the m lines?*

Answer

Claim: the algorithm never returns a previous assignment instance.

Proof: Let $X = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{i,j}$.
The reducions will reduce X *by* $d \times n^2$*. The additions will increase the current total by* $m \times n \times d$*. Since* $m < n$ *the resulting total will be* $X - (n - m)n \times d < X$ *hence a new equivalent instance of the previous assignment problem.*

Augmenting Paths

Definition

An alternating path is a sequence $\{a_{r_1 c_1}, a_{r_2 c_2}, \ldots, a_{r_k c_k}\}$ *of entries in the current assignment matrix such that:*

Augmenting Paths

Definition

An alternating path is a sequence $\{a_{r_1 c_1}, a_{r_2 c_2}, \ldots, a_{r_k c_k}\}$ *of entries in the current assignment matrix such that:*

Augmenting Paths

Definition

An alternating path is a sequence $\{a_{r_1c_1}, a_{r_2c_2}, \ldots, a_{r_kc_k}\}$ *of entries in the current assignment matrix such that: 1.* $a_{r_jc_j} = 0\ \forall j$
2. $c_{j+1} = c_j$ *if* j *is odd and* $r_{j+1} = r_j$ *if* j *is even.*
3. if $|i - j| > 1$ *then* $r_i \neq r_j$ *and* $c_i \neq c_j$.
4. The zeros on the path start with a non-selected zero and alternate between selected zeros and non-selected zeros.

Augmenting Paths

Definition

An alternating path is a sequence $\{a_{r_1 c_1}, a_{r_2 c_2}, \ldots, a_{r_k c_k}\}$ *of entries in the current assignment matrix such that: 1.* $a_{r_j c_j} = 0 \ \forall j$
2. $c_{j+1} = c_j$ *if* j *is odd and* $r_{j+1} = r_j$ *if* j *is even.*
3. if $|i - j| > 1$ *then* $r_i \neq r_j$ *and* $c_i \neq c_j$*.*
4. The zeros on the path start with a non-selected zero and alternate between selected zeros and non-selected zeros.

Definition

If there is no selected zero on column number c_k *then the path is an augmenting path.*

Augmenting Paths

Definition

An alternating path is a sequence $\{a_{r_1 c_1}, a_{r_2 c_2}, \ldots, a_{r_k c_k}\}$ *of entries in the current assignment matrix such that: 1.* $a_{r_j c_j} = 0 \; \forall j$
2. $c_{j+1} = c_j$ *if* j *is odd and* $r_{j+1} = r_j$ *if* j *is even.*
3. if $|i - j| > 1$ *then* $r_i \neq r_j$ *and* $c_i \neq c_j$*.*
4. The zeros on the path start with a non-selected zero and alternate between selected zeros and non-selected zeros.

Definition

If there is no selected zero on column number c_k *then the path is an augmenting path.*

Augmenting Paths

Definition

An alternating path is a sequence $\{a_{r_1c_1}, a_{r_2c_2}, \ldots, a_{r_kc_k}\}$ *of entries in the current assignment matrix such that: 1.* $a_{r_jc_j} = 0\ \forall j$
2. $c_{j+1} = c_j$ *if j is odd and* $r_{j+1} = r_j$ *if j is even.*
3. if $|i - j| > 1$ *then* $r_i \neq r_j$ *and* $c_i \neq c_j$*.*
4. The zeros on the path start with a non-selected zero and alternate between selected zeros and non-selected zeros.

Definition

If there is no selected zero on column number c_k *then the path is an augmenting path.*

Note that if we find an augmenting path, we can get a larger set of independent zeros by removing the independent zeros along the augmenting path and replacing them by the other zeros.

(Augmenting paths)

1. Once we find a set of independent zeros we can try to augment it. If we fail, we will be able to find a set of lines that covers all zeros whose size is equal to the number of independent zeros (that such a set of lines exists will be proved later).

(Augmenting paths)

1. Once we find a set of independent zeros we can try to augment it. If we fail, we will be able to find a set of lines that covers all zeros whose size is equal to the number of independent zeros (that such a set of lines exists will be proved later).

(Augmenting paths)

1. Once we find a set of independent zeros we can try to augment it. If we fail, we will be able to find a set of lines that covers all zeros whose size is equal to the number of independent zeros (that such a set of lines exists will be proved later).

2. Starting with a zero on a row with no independent zero (such a zero exists as every row contains zeros and there are only $m < n$ independent zeros), we build an alternating path.

(Augmenting paths)

1. Once we find a set of independent zeros we can try to augment it. If we fail, we will be able to find a set of lines that covers all zeros whose size is equal to the number of independent zeros (that such a set of lines exists will be proved later).

2. Starting with a zero on a row with no independent zero (such a zero exists as every row contains zeros and there are only $m < n$ independent zeros), we build an alternating path.

3. If the path ends in a selected independent zero, mark the column as essential.

(Augmenting paths)

1. Once we find a set of independent zeros we can try to augment it. If we fail, we will be able to find a set of lines that covers all zeros whose size is equal to the number of independent zeros (that such a set of lines exists will be proved later).

2. Starting with a zero on a row with no independent zero (such a zero exists as every row contains zeros and there are only $m < n$ independent zeros), we build an alternating path.

3. If the path ends in a selected independent zero, mark the column as essential.

4. Continue searching for alternating paths avoiding the essential columns. If you find an augmenting path *use it to get a larger set of independent zeros.*

(Augmenting paths)

1. Once we find a set of independent zeros we can try to augment it. If we fail, we will be able to find a set of lines that covers all zeros whose size is equal to the number of independent zeros (that such a set of lines exists will be proved later).

2. Starting with a zero on a row with no independent zero (such a zero exists as every row contains zeros and there are only $m < n$ independent zeros), we build an alternating path.

3. If the path ends in a selected independent zero, mark the column as essential.

4. Continue searching for alternating paths avoiding the essential columns. If you find an augmenting path use it to get a larger set of independent zeros.

5. The essetial columns plus all the rows that contain zeros not on essntial columns will be a set of m lines that covers all zeros.

Summary

(Final steps)

1. When you found m lines that cover all zeros, you completed step number A_2 of the Hungarian Method.

Summary

(Final steps)

1. When you found m lines that cover all zeros, you completed step number A_2 of the Hungarian Method.

Summary

(Final steps)

1. When you found m lines that cover all zeros, you completed step number A_2 of the Hungarian Method.

2. We go now to step number A_3 and repeat until we find a 0-cost assignment.

Summary

(Final steps)

1. When you found m lines that cover all zeros, you completed step number A_2 of the Hungarian Method.

2. We go now to step number A_3 and repeat until we find a 0-cost assignment.

(Summary)

What remains to be done is to prove the correctness of the assertions in the algorithm.

Summary

Summary

- *1. We proved that the reductions never cycle back to a previous instance. Furthermore, every time we reach step number A_6 the sum of all costs is reduced. This cannot go on for ever. The only reason for it to stop is when $m = n$ or we found a 0 cost assignment.*

Summary

- *1. We proved that the reductions never cycle back to a previous instance. Furthermore, every time we reach step number A_6 the sum of all costs is reduced. This cannot go on for ever. The only reason for it to stop is when $m = n$ or we found a 0 cost assignment.*
- *2. We still need to prove that if m is the maximum size of an independent set then there is a set of m lines that covers all zeros.*

Summary

- *1. We proved that the reductions never cycle back to a previous instance. Furthermore, every time we reach step number A_6 the sum of all costs is reduced. This cannot go on for ever. The only reason for it to stop is when $m = n$ or we found a 0 cost assignment.*
- *2. We still need to prove that if m is the maximum size of an independent set then there is a set of m lines that covers all zeros.*
- *3. We need to analyze the execution time of this algorithm as a function of n.*

Summary

- *1. We proved that the reductions never cycle back to a previous instance. Furthermore, every time we reach step number A_6 the sum of all costs is reduced. This cannot go on for ever. The only reason for it to stop is when $m = n$ or we found a 0 cost assignment.*
- *2. We still need to prove that if m is the maximum size of an independent set then there is a set of m lines that covers all zeros.*
- *3. We need to analyze the execution time of this algorithm as a function of n.*
- *4. We would like to find the appropriate mathematical tools to deal with this and similar problems.*