# CS3933 Fall 2007 – Group Assignment 5
# Due Wednesday November 28, 2007, at 5pm.
# Sample Solution

1. (8 marks; see options) Design and write a dynamic programming algorithm that, given a sequence A[1..n] of integers, finds the segment (A[i..j]) whose sum is the largest.

   Asymptotically analyse the running time of your algorithm.

   Two possibilities:

   Option 1 (worth 8 marks: 6 for the algorithm, 2 for the analysis)

```
sum ← A[1]
ci ← 1
max ← sum
mi ← 1
mj ← 1
for i from 2 to n
    sum ← sum + A[i]
    if sum < A[i]
            sum ← A[i]
            ci ← i
    if max < sum
            max ← sum
            mi ← ci
            mj ← j
return i, j
```

   Time analysis:
   The for loop iterates $n - 1$ times. The rest of the algorithm contributes a constant factor, so the algorithm as a whole takes time $\in \Theta(n)$.

Option 2 (worth 6 marks: 4 for the algorithm, 2 for the analysis)

Let $M[1..n][1..n]$ be a 2-dimensional array of integers.

```
max ← A[1]
mi ← 1
mj ← 1
for i from 1 to n
    M[i][i] ← A[i]
    for j from i + 1 to n
        M[i][j] ← M[i][j − 1] + A[j]
        if max < M[i][j]
            max ← M[i][j]
            mi ← i
            mj ← j
return i, j
```

Time analysis:

Upper bound: the outer for loop iterates $n$ times. For each of these iterations, the inner for loop iterates at most $n-1$ times. The rest of the algorithm contributes at most a constant factor, so the algorithm as a whole runs in time $\in O(n^2)$.

Lower bound: consider the iterations of the outer loop where $i \leq \frac{n}{2}$. There are at least $\frac{n}{2} - 1$ such iterations. For each of these iterations, the inner for loop (on $j$) has at least $\frac{n}{2}$ iterations, for $j$ such that $\frac{n}{2} < j \leq n$. Therefore the algorithm runs in time $\in \Omega((\frac{n}{2} - 1) \cdot \frac{n}{2}) \;=\; \Omega(n^2)$.

Since the algorithm runs in time $\in O(n^2)$ and $\in \Omega(n^2)$, it runs in time $\in \Theta(n^2)$.

2. (worth 18 marks; deduct 4 marks if the answers to a and b are reversed)
When we initially looked at class scheduling, each class consisted of one time interval. However, consider a situation of scheduling courses that can consist of many nonoverlapping time intervals; in such a situation, in order to schedule (or pick) a course, we have to include all of its intervals.

Given a set of courses, where each course is a set of time intervals $(s, f)$, and a positive integer $k$, we want to determine if there is a set of $k$ courses that can all be scheduled in the same room, without any overlapping intervals. This is the Course Scheduling problem.

Show that Course Scheduling is equivalent to Independent Set, by:

(a) (5 for transformation, 4 for proof that it works)
Proving that COURSE SCHEDULING $\leq_p$ INDEPENDENT SET.

Let $\mathcal{C} = \{C_i\}$, where each $C_i = \{(s_j, f_j)\}$, and $k_{CS}$ be any input instance of COURSE SCHEDULING.

Construct an instance of INDEPENDENT SET, $G = (V, E)$ and $k_{IS}$, by:
Let $k_{IS} = k_{CS}$.
Let $V = \{i \mid \text{such that } C_i \in \mathcal{C}\}$
Let $E = \{(u, v) \mid \exists (s_x, f_x) \in C_u \text{ and } (s_y, f_y) \in C_v \text{ such that } s_x \leq s_y < f_x\ \}$.

This transformation can be constructed by checking all pairs of intervals for all pairs of courses, and so is a polynomial time transformation.

Need to show that $\mathcal{C}$, $k_{CS}$ is a yes-instance of COURSE SCHEDULING if and only if $G = (V, E)$, $k_{IS}$ is a yes-instance of INDEPENDENT SET.

$\Rightarrow$: Let $\mathcal{C} = \{C_i\}$ and $k_{CS}$ be any yes-instance of COURSE SCHEDULING. Then there is a collection of courses $\mathcal{S} \subseteq \mathcal{C}$, with $|\mathcal{S}| = k$, such that for all $C_i, C_j \in \mathcal{S}$, and all $(s_x, f_x) \in C_i$ and $(s_y, f_y) \in C_j$, intervals $(s_x, f_x)$ and $(s_y, f_y)$ do not overlap (so $s_x \leq s_y \Rightarrow f_x \leq s_y$).

Since none of the intervals in $C_i$ and $C_j$ ($\in \mathcal{S}$) overlap, then by the construction of $G$, $(i, j) \notin E$. If we pick $V' \subseteq V$ such that $u \in V'$ if and only if $C_u \in \mathcal{S}$, then $\forall u, v \in V'$, $(u, v) \notin E$, making $V'$ an independent set. Since $|V'| = |\mathcal{S}| = k_{CS} = k_{IS}$, then $V'$ is an independent set in $G$ of at least $k_{IS}$ vertices. Therefore $G = (V, E)$, $k_{IS}$ is a yes-instance of INDEPENDENT SET.

$\Leftarrow$: Let $G = (V, E)$ and $k_{IS}$ be a yes-instance of INDEPENDENT SET that is produced by the transformation. Then there is a set $V' \subseteq V$ with $|V'| \geq k_{IS}$, such that for all $u, v \in V'$, $(u, v) \notin E$.

Since each $u \in V$ corresponds to a course $C_u$, then by the construction of $G$, for all $u, v \in V'$, since $(u, v) \notin E$, the courses $C_u$ and $C_v$ do not contain any overlapping intervals. The collection of courses $\mathcal{S} = \{C_u \mid u \in V'\}$ is thus a collection of courses that do not overlap. $|\mathcal{S}| = |V'| \geq k_{IS} = k_{CS}$, and since there is thus a nonoverlapping collection of size at least $k_{CS}$, there is a nonoverlapping collection of size $k_{CS}$. Therefore the original $\mathcal{C}$, $k_{CS}$ is a yes-instance of COURSE SCHEDULING.

Therefore INDEPENDENT SET $\leq_p$ COURSE SCHEDULING.

(b) (5 for transformation, 4 for proof that it works)
Proving that INDEPENDENT SET $\leq_p$ COURSE SCHEDULING.

Given any instance of INDEPENDENT SET, $G = (V, E)$ and $k_{IS}$
Construct an instance of COURSE SCHEDULING by:

Let $k_{CS} = k_{IS}$
Let $\mathcal{C} = \{C_u \mid u \in V\}$
Number the edges of $G$ from 1 to $|E|$, so that $n(u, v) =$ the unique number of the edge.
Let $C_u\{(n(u, v), n(u, v) + 1) \mid (u, v) \in E\ \}$.

This transformation can be constructed by numbering the edges and then traversing the graph, so it is a polynomial time transformation.

We need to show that $G, k_{IS}$ is a yes-instance of Independent Set if and only if $\mathcal{C}, k_{CS}$ is a yes-instance of Course Scheduling.

$\Rightarrow$: Let $G = (V, E), k_{IS}$ be any yes-instance of Independent Set. Then there is a subset of vertices $V' \subseteq V$, $|V'| \leq k$, such that $\forall u, v \in V'$, $(u, v) \notin E$.

By the construction of $\mathcal{C}$, an interval $(i, j)$ occurs in a course $C_u$ if and only if $j = i + 1$ and edge number $i$ is incident on $C_u$. Thus for two courses $C_u$ and $C_v$ to overlap, they must contain the same interval $(i, i + 1)$, and this interval corresponds to edge $i$ that would be incident on both $u$ and $v$.

Let $\mathcal{S} = \{C_u \mid u \in V'\}$. For any pair of vertices $u, v \in V'$, there is no edge $(u, v) \in E$. Since there is thus no edge incident on both $u$ and $v$, there will not be an interval $(i, i + 1)$ that is common to both $C_u$ and $C_v$, and since those are the only types of intervals and overlaps constructed, $C_u$ and $C_v$ will not have any overlapping intervals. Therefore $\mathcal{S}$ is a nonoverlapping set of courses, and since $|\mathcal{S}| = |V'| \geq k_{IS} = k_{CS}$, there is a non-overlapping set of $k_{CS}$ courses. Therefore the constructed $\mathcal{C}, k_{CS}$ is a yes-instance of Course Scheduling.

$\Leftarrow$: Let $\mathcal{C} = \{C_u \mid u \in V\}$ (as constructed) and $k_{CS}$ be any yes-instance of Course Scheduling produced by the transformation. Then there is a set $\mathcal{S} \subseteq \mathcal{C}$ of courses such that $\forall C_u, C_v \in \mathcal{S}$, $C_u$ and $C_v$ do not contain any overlapping intervals.

Since $C_u$ and $C_v$ (both $\in \mathcal{S}$) do not overlap, then they do not contain a common interval $(i, i + 1)$. By the construction of $\mathcal{C}$, if there was an edge $(u, v) \in E$, then there would be an interval $(i, i + 1)$ representing this edge present in both $C_u$ and $C_v$. Since there is no such interval common to $C_u$ and $C_v$, there cannot be an edge $(u, v) \in E$.

Let $V' = \{u \mid C_u \in \mathcal{S}\}$. Then $\forall u, v \in V'$, $C_u, C_v \in \mathcal{S}$, so $(u, v) \notin E$. Thus $V'$ is an independent set of size $|V'| = |\mathcal{S}| = k_{CS} = k_{IS}$ in $G$. Therefore $G$, $k_{IS}$ is a yes-instance of Independent Set.

Therefore Course Scheduling $\leq_p$ Independent Set.

Note: Since it is very important that you understand how to construct the correct

direction of a reduction, some marks will be deducted if you confuse (a) and (b). Clarify the direction with the professor if you are unsure.