CS524
HW#4

DUE: Monday, November 14

1. (4 points) Suppose you want to satisfy many clauses of an instance of 3-SAT. Show that for any instance of 3-SAT there is an interpretation that satisfies at least 7/8 of the clauses. That is, if the instance has m 3-clauses, then there is an interpretation that satisfies $\lceil 7m/8 \rceil$ of them.

2. (20 points) Assume that n people want to schedule intervals of time to use a computer, where the computer can not be shared at any time among two or more people. Thus, the people's requests are $\rho = \{(s_1, f_1), ..., (s_n, f_n)\}$, $s_i < f_i, 1 \le i \le n$. Intuitively, if $(s_i, f_i) = (10, 22)$, this means that person i wants to use the computer for the time interval running from time s_i=10 to time f_i=22. As in life, we assume that people's wants/needs are inflexible; they are not willing to negotiate. Requests (s_i, f_i) and (s_j, f_j) are *compatible* if $f_i \le s_j$ or $f_j \le s_i$, and a set of requests is *feasible* if they are pairwise compatible. An *optimal schedule* is a maximum feasible subset of ρ.

a Consider the following algorithm:

GREEDY1
$\Gamma \leftarrow \emptyset$
while there are requests in ρ which are compatible with each request in Γ **do**
  Select (s_i, f_i) which is compatible with each request in Γ and minimizes $f_i - s_i$
  $\Gamma \leftarrow \Gamma \cup \{(s_i, f_i)\}$
  $\rho \leftarrow \rho - \{(s_i, f_i)\}$

Clearly this algorithm computes a feasible set of requests Γ. Either prove that GREEDY1 always produces an optimal schedule, or show that it can fail.

b Consider the following algorithm:

GREEDY2

$\Gamma \leftarrow \varnothing$

while there are requests in ρ which are compatible with each request in Γ **do**

Select (s_i, f_i) which is compatible with each request in Γ and minimizes s_i

$\Gamma \leftarrow \Gamma \cup \{(s_i, f_i)\}$

$\rho \leftarrow \rho - \{(s_i, f_i)\}$

Clearly this algorithm computes a feasible set of requests Γ. Either prove that GREEDY2 always produces an optimal schedule, or show that it can fail.

c Consider the following algorithm:

GREEDY3

$\Gamma \leftarrow \varnothing$

while there are requests in ρ which are compatible with each request in Γ **do**

Select (s_i, f_i) which is compatible with each request in Γ and minimizes f_i

$\Gamma \leftarrow \Gamma \cup \{(s_i, f_i)\}$

$\rho \leftarrow \rho - \{(s_i, f_i)\}$

Clearly this algorithm computes a feasible set of requests Γ. Either prove that GREEDY3 always produces an optimal schedule, or show that it can fail.

In real life scheduling problems, some jobs are more important than others. In the *optimal weighted scheduling problem*, the input is $\rho_w = \{(w_1, s_1, f_1), \ldots, (w_n, s_n, f_n)\}$, $w_i \geq 0$ for all $1 \leq i \leq n$, and an optimal schedule is a feasible schedule $\Gamma_w = \{(w_{i_1}, s_{i_1}, f_{i_1}), \ldots, (w_{i_j}, s_{i_j}, f_{i_j})\}$ which maximizes $\sum_{1 \leq k \leq j} w_{i_k}$ over all feasible schedules. That is, it is a feasible schedule which maximizes the sum of the weights of the scheduled requests. You may assume that $f_1 \leq f_2 \leq \ldots \leq f_n$.

d Describe a linear (in *n*) time algorithm to compute a solution to the *optimal weighted scheduling problem.*

C.S.524

SOLUTION FOR H.W. #4

1. With each of the m 3-clauses we associate a random variable $X_i, 1 \le i \le m$ which is 1 if an interpretation satisfies the i^{th} clause and 0 otherwise. For a random interpretation (which chooses values 0 or 1 with equal probability for each variable), the expected value of X_i is 7/8. We note that rv $X = \sum_{1 \le i \le m} X_i$ is the number of 3-clauses which are satisfied by an interpretation. By Linearity of Expectation,

$E[X] = E\left[\sum_{1 \le i \le m} X_i\right] = \sum_{1 \le i \le m} E[X_i] = \sum_{1 \le i \le m} 7/8 = 7m/8$, and there must be some interpretation which satisfies at least this many clauses. Because a clause is either satisfied or not satisfied, there must be an interpretation which satisfies at least $\lceil 7m/8 \rceil$ 3-clauses.

2. ***a*** GREEDY1 fails on $\rho = \{(0,5),(4,6),(5,10)\}$. It produces the schedule {(4, 6)}, but the optimal schedule is {(0, 5), (5, 10)}.
b GREEDY2 fails on $\rho = \{(0,10),(1,2),(3,4)\}$. It produces the schedule {(0, 10)}, but the optimal schedule is {(1, 2), (3, 4)}.
c GREEDY3 always succeeds. To see this, let $\Gamma = \{(s_{i_1}, f_{i_1}),\ldots,(s_{i_l}, f_{i_l})\}$ be the schedule constructed by GREEDY3, and assume that it's ordered by the finish times, that is, $f_{i_k} \le f_{i_{k+1}}$ for $1 \le k < l$. Let $\Pi = \{(s_{j_1}, f_{j_1}),\ldots,(s_{j_m}, f_{j_m})\}$ be any other set of intervals ordered by the finish times. We prove two claims.

CLAIM 1: $f_{i_r} \le f_{j_r}$ for $1 \le r \le l$.

PROOF OF CLAIM1: The CLAIM is clearly true for r=1. Assume that $f_{i_{r-1}} \le f_{j_{r-1}}$. When $(s_{i_{r-1}}, f_{i_{r-1}})$ is added to Γ, both (s_{i_r}, f_{i_r}) and (s_{j_r}, f_{j_r}) are still in Γ. Since GREEDY3 chooses (s_{i_r}, f_{i_r}) to add to Γ, it follows that $f_{i_r} \le f_{j_r}$, establishing the CLAIM.

CLAIM 2: Γ is optimal.

PROOF OF CLAIM 2: Assume Γ is not optimal. Then m>l. By CLAIM 1, $f_{i_l} \le f_{j_l}$. But because Π is feasible, $f_{j_l} \le s_{j_{l+1}}$. But this is impossible because GREEDY 3 would have added $(s_{j_{l+1}}, f_{j_{l+1}})$ to Γ, which yields a contradiction.

d For $\rho_w = \{(w_1, s_1, f_1), ..., (w_n, s_n, f_n)\}$, we let $\Phi(\rho_w)$ denote the maximum $\sum_{1 \le k \le j} w_{i_k}$ over all feasible schedules, and we let $\Delta(\rho_w)$ denote ρ_w with (w_n, s_n, f_n) removed as well as all triples (w_i, s_i, f_i) such that $f_i > s_n$, that is, all triples (w_i, s_i, f_i) which are not compatible with (w_n, s_n, f_n). To derive $\Phi(\rho_w)$, we note that either (w_n, s_n, f_n) **is** included in an optimal schedule, in which case $\Phi(\rho_w) = w_n + \Phi(\Delta(\rho_w))$ or it **is not**, in which case $\Phi(\rho_w) = \Phi(\rho_w - \{(w_n, s_n, f_n)\})$.

$$\Phi(\rho_w) = \begin{cases} 0, if\ \rho_w = \varnothing \\ \max\left(w_n + \Phi(\Delta(\rho_w)), \Phi\left(\rho_w - \{(w_n, s_n, f_n)\}\right)\right), \text{otherwise} \end{cases}$$

As a program, having precomputed $\Delta(i)$ for all *i*,

$\Phi[0] \leftarrow 0$

for $i \leftarrow 1$ **to** n **do**

$\quad \Phi[i] \leftarrow \max\left(w_n + \Phi[\Delta(i)], \Phi[i-1]\right)$