Improvements on a Time Slot Allocation Algorithm

Lewin Gan, Allen Xiao, Edwin Liao

CS270, Spring 2013
University of California, Berkeley

May 9, 2013

Motivation

Motivation

- Schedule tutors into time/location slots
  - HKN's undergrad tutoring hours

Motivation

- Schedule tutors into time/location slots
  - HKN's undergrad tutoring hours
- Old implementation:
  - Linear programming to find a feasible matching
  - Unoptimized hill-climbing to improve solution
  - Ran overnight to get satisfactory solution

Motivation

- Schedule tutors into time/location slots
  - HKN's undergrad tutoring hours
- Old implementation:
  - Linear programming to find a feasible matching
  - Unoptimized hill-climbing to improve solution
  - Ran overnight to get satisfactory solution
- *General Assignment Problem* (GAP)
  - n agents and m tasks
  - Each agent has a max capacity for tasks
  - Minimize the total cost of assigning all tasks
- Maximum General Assignment Problem
  - Still misses some parts of the model

Features

Features

- Slots have preferences on:
  - Courses tutored at that slot's location

Features

- Slots have preferences on:
    - Courses tutored at that slot's location
- Tutors have preferences on:
    - Courses they are confident in
    - Times, locations
    - Whether their assigned slots are adjacent

Features

- Slots have preferences on:
    - Courses tutored at that slot's location
- Tutors have preferences on:
    - Courses they are confident in
    - Times, locations
    - Whether their assigned slots are adjacent
- Hard to encode the last preference without making the problem exponential in size

A New Hope

A New Hope

- Assume $|S| \leq k|T|$, where k is the number of slots a tutor needs to take

A New Hope

- Assume $|S| \leq k|T|$, where k is the number of slots a tutor needs to take
- Then each tutor will have exactly k slots, and each slot has at least one tutor

A New Hope

- Assume $|S| \leq k|T|$, where k is the number of slots a tutor needs to take
- Then each tutor will have exactly k slots, and each slot has at least one tutor
- Then, while we have remaining slots:

A New Hope

- Assume $|S| \leq k|T|$, where k is the number of slots a tutor needs to take
- Then each tutor will have exactly k slots, and each slot has at least one tutor
- Then, while we have remaining slots:
    - Use maximum weight matching to find an assignment from tutors to slots remaining

A New Hope

- Assume $|S| \leq k|T|$, where k is the number of slots a tutor needs to take
- Then each tutor will have exactly k slots, and each slot has at least one tutor
- Then, while we have remaining slots:
    - Use maximum weight matching to find an assignment from tutors to slots remaining
    - Remove the slots that were matched ($|T|$ of them)

A New Hope

- Assume $|S| \leq k|T|$, where k is the number of slots a tutor needs to take
- Then each tutor will have exactly k slots, and each slot has at least one tutor
- Then, while we have remaining slots:
  - Use maximum weight matching to find an assignment from tutors to slots remaining
  - Remove the slots that were matched ($|T|$ of them)
- This will run at most k times

Why this Approach?

Why this Approach?

- Weights change depending on what is already matched
    - (i.e. Tutor prefers slots that are adjacent)

Why this Approach?

- Weights change depending on what is already matched
  - (i.e. Tutor prefers slots that are adjacent)
- Some matchings infeasible
  - (Tutors can not go to two slots that are at the same time)

Why this Approach?

- Weights change depending on what is already matched
    - (i.e. Tutor prefers slots that are adjacent)
- Some matchings infeasible
    - (Tutors can not go to two slots that are at the same time)
- Iterations allow us to change weights in between

Effectiveness

Effectiveness

- At each iteration, we choose a subset of slots such that the weights matching with tutors is optimal
  - Let the first iteration have value X

Effectiveness

- At each iteration, we choose a subset of slots such that the weights matching with tutors is optimal
  - Let the first iteration have value X
- Then, any subset of $|T|$ slots maximally matched with the tutors will have value $\leq X$

Effectiveness

- At each iteration, we choose a subset of slots such that the weights matching with tutors is optimal
  - Let the first iteration have value X
- Then, any subset of $|T|$ slots maximally matched with the tutors will have value $\leq X$
- Worst case: algorithm will only find one such subset with value X, and other subsets with value zero; optimal could have all subsets with value X
  - Our algorithm gives solution of X
  - Optimal value is kX

Effectiveness

- At each iteration, we choose a subset of slots such that the weights matching with tutors is optimal
  - Let the first iteration have value X
- Then, any subset of $|T|$ slots maximally matched with the tutors will have value $\leq X$
- Worst case: algorithm will only find one such subset with value X, and other subsets with value zero; optimal could have all subsets with value X
  - Our algorithm gives solution of X
  - Optimal value is kX
- Thus, we have a k-approximation (in our case, $k = 2$)

Thanks for your time.

Any questions?