Resource Allocation Algorithms

Lecture 12

Lecture by Dr. Tami Tamir
Typeset by Steven Karas

2018-06-19
Last edited 18:03:11 2018-06-19

Disclaimer These lecture notes are based on the lecture for the course Resource Allocation Algorithms, taught by Dr. Tami Tamir at IDC Herzliyah in the spring semester of 2017/2018. Sections may be based on the lecture slides written by Dr. Tami Tamir.

1 Recitation

The exam is next Monday, we will only cover example problems and their solution.

1.1 Packing 2 - slide 31

Given an instance for the Bin-Packing problem. Let OPT be the number of bins needed by the optimal solution (in which all bins have items totalling to 1). Assume that we can pack into each bin items totalling $\frac{4}{3}$. Suggest an efficient[1] algorithm that uses at most OPT bins.

1.1.1 Answer

Consider the LPT algorithm for $p||C_{\max}$, in which we assign jobs to run with minimal makespan on p machines. If we were given OPT, then we could just run LPT and if it's possible to pack items into OPT bins of size 1, then we can schedule those same jobs on OPT machines with a makespan between 1 and $\frac{4}{3}$, because LPT is a $frac43$-approximation.

So our algorithm is to run a binary search to find the minimal m such that all items are packed by LPT on m machines. The proof follows from the approximation ratio.

1.2 Facility Location 1

Consider the 2-approximation for k-center facility location:

2-approximation for k-center

```
Choose the first center arbitrarily
Until there are $k$ centers:
  Choose the vertex that is furthest from the current centers
```

For any given n_0, describe a graph with $n \geq n_0$ vertices, such that **every** run of the algorithm with $k = 1$ and $k = 2$ is optimal, and every run with $k = 3$ provides a 1.5-approximation. Describe the graph, an optimal solution, the algorithms' solution, and conclude the approximation ratio.

1.2.1 Answer

Given n_0, let n be a multiple of 12 at least n_0. Consider a cycle of n nodes. Any node is an optimal 1-center (max distance $=n/2$). Any two nodes that are $n/2$ apart from the other forms an optimal 2-center (max distance $=n/4$). An optimal 3-center consists of three nodes that are $n/3$ apart (max distance $=n/6$). However, the third center of the algorithm does not affect the max distance (which remains $n/4$).

[1] Missing from the original, but otherwise bruteforce is the easy answer

1.3 Scheduling 4 - slide 9

Let C^* be the value of an optimal solution for the problem $p|\text{pmtn}|C_{\max}$ with $m > 1$ machines and $n > m$ jobs. Assume that all jobs lengths are increased by one, that is, for every job $p'_j = p_j + 1$. Denote by C'^* the value of an optimal solution for $p|\text{pmtn}|C_{\max}$ on the resulting instance.

What is the minimal value of $C'^* - C^*$?

1.3.1 Answer

We know the makespan for this problem in closed form. Due to the change, the total processing increased by n, and so the maximum makespan increases by at least 1.

1.4 Scheduling 2 - slide 5

Let I be an instance of $n = 20$ jobs for which an optimal solution of $P_2||C_{\max}$ has value 100.

Let I' be an instance derived from I by increasing by 1 the processing times of all 20 jobs.

Prove or disprove the following:

1. The solution of $P_2||C_{\max}$ for I' has value at least 110.
2. The solution of $P_2||C_{\max}$ for I' has value at most 110.
3. The solution of $P_2||C_{\max}$ for I' has value at most 119.
4. If the solution of $P_2||C_{\max}$ for I' has value exactly 110, then there are exactly 10 jobs on each machine in both optimal solutions.

1.4.1 Solutions

1. False. Let 1 job take 100, the rest take 1.
2. False. Let 1 job take 100, 1 take 82, and the rest take 1. This maximizes the diff on the second machine. The total makespan must be over 110 by pigeonhole on the partition over total sizes (220).
3. True. I missed the reason why though.
4. False. Take 99, 9, 1, 17 times 91/17... this leads to both machines having 110, but not 10 jobs each.

1.5 Facility Location 4

Given a weighted graph in which the vertex v is a vertex 1-center, and the vertex u is a vertex 1-median.

Prove or disprove (one is true, the other is not):

1. If we increase the weight of each edge by 1, then v is still a 1-center.
2. If we double the weight of each edge, then u is still a 1-center.

1.5.1 Answer

1. False. Counterexample is a line graph with the edge between one of the endpoints having weight n.
2. True. Proof follows arithmetically from the objective function.

1.6 Scheduling 3

Minimum makespan on two machines, flowshop scheduling, consisting of n jobs. It is known that for a job j, then $a_j = b_j = 2^j$. If one of the machines has to be replaced and it will take twice as long for jobs. Which machine should be replaced, and why? Does it matter? What's the damage?

1.6.1 Answer

Doesn't matter. If the first machine is replaced, then for all jobs $a_j > b_j$, and by Johnson rule the optimal solution is to process the jobs according to the decreasing b-order, resulting in makespan $2^{n+2} - 1$. If the second machine is replaced, then for all jobs $a_j < b_j$, and by Johnson rule the optimal solution is to process the jobs according to increasing a-order, resulting in makespan $2^{n+2} - 1$.

1.7 Algorithmic Game Theory 3 - Slide 26

Given a graph of two disjoint paths from s to t, where both paths are greater than 1. Denote w.l.o.g. x as the length of the shorter path, and y as the length of the greater path. We are given that all agents will bid 1 ($b(e) = 1$).

1. How much will a VCG mechanism pay for a feasible solution?
2. Denote $y - x = a$. Does the frugality ratio of the VCG grow as a increases? Explain why your answer justifies this metric.

1.7.1 Answer

As per VCG, we will select the shorter path and pay it:

$$\underbrace{x}_{\text{number of vertices in the path}} \left(1 + \underbrace{y - x}_{\text{TODO}}\right)$$

The frugality ratio appeared, but I didn't catch it.

1.8 Algorithmic Game Theory 3 - Slide 28

I missed this question.

1.9 Scheduling 5

We solve the problem $1|r_j|\sum_j T_j$ using branch and bound. In each level, we schedule another job in processing order (not from finish to start), while consdiering release times. In other words, the first level has an empty schedule, and at level 1, there are n schedules, where the schedule j has a single job j that starts at r_j. The bound is computed by taking the minimal tardiness for the given partial schedule.

For the given instance, complete B&B, and explain each of the steps:

j	**p**	**r**	**d**
1	5	0	12
2	3	1	8
3	4	2	7

1.9.1 Answer

The root has three children:

P1 J1 is not late, and completes at $r_1 + p_1 = 0 + 5 = 5$. One of $J2, J3$ completes at 12, meaning a delay of at least 4.

P2 J2 is not late, and completes at $r_2 + p_2 = 1 + 3 = 4$. ...

P3 J3 is not late, and completes at $r_3 + p_3 = 2 + 4 = 6$. ...

In the end, we find that P231 is the optimal.

1.10 Scheduling 1

n jobs need to be processed by two machines. M_1 can process any number of jobs simultaneously. M_2 can only process one job at a time. Job j needs to be processed for a_j and b_j time on M_1, M_2 respectively. Suggest an algorithm for minimizing the makespan under the following constraints:

1. When each job must be processed by M_1 before it can be processed by M_2.
2. When each job must be processed by M_2 before it can be processed by M_1.

1.10.1 1 then 2

Trivially, process everything on M_1 all at once, and then solve the $1|r_j|C_{\max}$ problem where the release time for job j is a_j. Greedy works, and proof follows by exchange argument over idle times on M_2. Full proof can be found on the slide[2].

[2] This proof is what is expected as a full proof for the exam with an explanation

1.10.2 2 then 1

Process the jobs on M_2 in decreasing order of a_j. Proof sketch is that we want to minimize the makespan, which is constant for M_2, and its extension on M_1. Full proof follows by exchange argument over makespan.

1.11 Packing 1

Present an instance for Bin-Packing for which First-Fit-Decreasing uses three bins, while First-Fit uses only two bins. Describe the items and the packings.

1.11.1 Answer

$$0.5, 0.3, 0.4, 0.2, 0.2, 0.2, 0.2$$

The sort breaks our efficient packing of the second bin. The reason this happens is because the

1.12 Facility Location 3 - Slide 18

Chinese postman problem for the given graph.

1. Find as a function of x, the minimal length of a Chinese postman cycle in this network. It is given that $w(E) = 185.5 + 3x$.
2. Assume that $x = 2$. Find the minimal length of a Chinese postman path that begins in F and ends in D.

1.12.1 Chinese postman cycle

The only vertices with an odd degree are A and E. The shortest path between these costs $\min(20 + 2x, 25.5)$.

1.12.2 Chinese postman from F to D

F and D have even degree, so we need to add extra edges to the graph. A and E have odd degree, so we need to change them also.

Create the clique with shortest paths to determine which paths to augment. The minimum matching has weight 28. Therefore the legnth of the CPP is $185.5 + 3x + 28 = 219.5$.

1.13 Facility Location 2

Consider the cover problem where we need to establish servers that must be within s_i of each vertex v_i. Given an instance and the knowledge that the optimal solution uses m servers. Increasing by 1 the length of all edges in the tree. The new optimal solution uses m' servers.

1. What is the maximal ratio? Explain.
2. What is the minimal ratio? Explain.

1.13.1 Maximal ratio

The maximal ratio is on order n. Consider a star graph with edge lengths 1 and all $s_i = 1$. A single facility in the middle gives us $m = 1$, while $m' = n$.

Cannot be more than n because of basic limits.

1.13.2 Minimal ratio

The minimal ratio is 1. Show an instance that isn't affected. It can't be lower, because it can't improve the solution.

1.14 Algorithmic Game Theory 2

In class we proved that LPT produces a NE in job scheduling games on identical machines. We will extend this result for machines with different speeds. Let s_i denote the speed of machine i. It means that it takes p_j/s_i time to process job j on M_i. The completion time of M_i is $C_i = \sum_{j \in i} p_j/s_i$. In our game, this is also the cost of each of the jobs assigned to M_i.

LPT sorts the jobs such that $p_1 \geq \ldots \geq p_n$.

Question 1: Let $m = 2$, $s_1 = 1$, $s_2 = 2$. Show that LPT of an instance with 8 jobs having lengths $\{12, 8, 8, 5, 3, 3, 2, 2\}$.

Question 2: Prove that for any m and n, LPT produces a NE. That is, after the assignment is over, no single job would like to migrate. Use induction on the number of jobs already assigned. Formally, show that for each $k = 1, \ldots, n$ after j_k is assigned, no job in $1, \ldots, k$ can improve by migrating.

1.14.1 Answer 1

The slow machine gets jobs of length $\{8, 3, 2\}$. The fast machine gets $\{12, 8, 5, 3, 2\}$. The makespan is on M_2. $C_2 = 30/???$

1.14.2 Answer 2

Base case: After the first job is assigned on the fastest machine, it will not benefit by migrating.

Step: Assume that after j_{k-1} is assigned, no job in $1, \ldots, k-1$ can improve by migrating. Consider the assignment of j_k. Assume that it is assigned to M. Clearly, j_k itself won't migrate as it is assigned by the algorithm on the machine minimizing its cost. Any job out of M won't migrate because the situation on M hasn't improved, and the other machines haven't changed. The only candidates that might want to migrate are those assigned to M.

Assume for contradiction that j_z where $z < k$ would benefit from migrating from M to M'. Let s, s' be the speeds of M, M'. Let C, C' denote the load on M, M' before j_k is assigned. j_k is assigned because $C + p_k/s \leq \ldots$...

2 Exam

The exam will be held at 15:45 on Monday. We can bring a single page formula sheet and a simple calculator. The exam will be largely technical, with some that are prove/disprove, some that are given something and change it, etc. There will be a choice in questions from the following:

- scheduling
- packing
- facility location

References

[1] Michael L Pinedo. *Scheduling: theory, algorithms, and systems.* Springer, 2016.