Notes on Advanced Optimization

40.302 Advanced Topics in Optimization, Term 7 2019

Yustynn Panicker

February 2, 2019

Contents

1 Meta

1.1 Professor

- Selin Damla Ahipasaoglu
- Room 1.702.03

1.2 Unofficial Project

Nothing official this term due to large class size. Can still do, and approach Selin for ideas

1.3 Grading

%	Component Name	Date
60%	Final Exam	W6
20%	Weekly Homeworks	Assigned Mon; Due next Thurs;
20%	Class and Lab Participation	-

2 TODO Questions

2.1 TODO Difference between Q_m and R_m?

Both seem to have machines that perform identical processing tasks, just at different speeds.

Phrased differently: difference between the following technical adjective for machines: `unrelated` and `uniform`

2.1.1 Reference from W1 Slides

- $\mathrm{Q_m}$: **m** uniform machines in parallel with different speeds
  Speed of machine i is v_i. Assuming that job j is completely processed on machine i, $p_{ij} = \frac{p_j}{v_i}$.
- $\mathrm{R_m}$: **m** unrelated machines in parallel
  Machine i processed the job j at speed v_{ij}. Then $p_{ij} = \frac{p_j}{v_{ij}}$.
  In this case, there is no particular relationship between processing times on different machines.

2.2 TODO Is `regular` a term for increasing functions in general?

2.3 TODO Why must active schedules be nonpreemptive?

Hypothesis: makes activity == non-delay, which trivializes the term `active`

3 W1: Introduction to Scheduling

3.1 Sugested Language

AMPL (A Mathematical Programming Language)

3.2 Why Scheduling?

- Very hard problem -> Good practice for opti/modelling

3.3 Definition of Scheduling

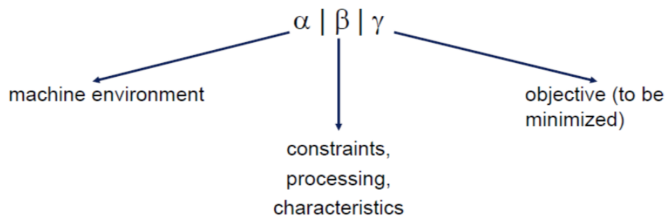
Deals with distributing a set of **jobs (tasks)** to **machines (processing units) over *time** subject to **constraints** with an objective to optimize ≥ 1 criteria

3.3.1 Assumptions

- Each machine has capacity of 1 job
- At each time unit, each job can be processed only on one machine

3.4 Terms and Notation

3.4.1 General



E.g. $J_m \mid\mid C_{max}$

Symbol	Meaning
$M = \{1, 2, \ldots, m\}$	Set of machines
$N = \{1, 2, \ldots, n\}$	Set of jobs

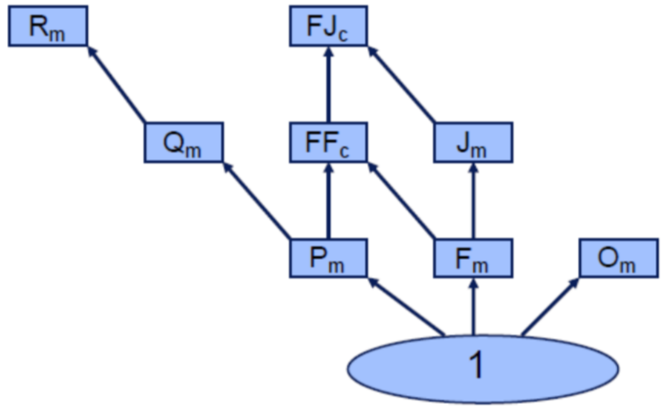
3.4.2 Jobs

Symbol	Meaning
p_{ij}	`Processing time` of job j on machine i
p_j	`Identical processing time` of job j on all machines
r_j	`Release date` of job j
d_j	`Due date` of job j; Penalty after d_j
$\bar{d}_j$	`Deadline` of job j; Completion after $\bar{d}_j$ not allowed
w_j	`Weight` of job j

3.4.3 Machine Environment (α)

Symbol	Term	Description/notes
1	Single machine	
P_m	m *identical* machines in parallel	Job j requires single operation, may be processed on any of the m machines
Q_m	m *uniform* machines in parallel with different speeds	Speed of machine i is v_i; $p_{ij} = \frac{p_j}{v_i}$
R_m	m unrelated machines in parallel	Speed of machine i is v_i; $p_{ij} = \frac{p_j}{v_i}$
F_m	`Flow shop` with m machines in series	Each job must be processed on every machine; predetermined route
FF_c	`Flexible flow shop` with c stages in series	Several identical machines per stage; Each job can only be processed on 1 machine per stage
J_m	`Job shop` with m machines	Jobs have predetermined routes
FJ_c	`Flexible job shop` with c stages	Multiple identical machines per stage
O_m	`Open shop` with m machines	Each job must be processed on every machine; No predetermined routes

3.4.3.1 Hierarchical Representation



3.4.4 Processing Details and Constraints (β)

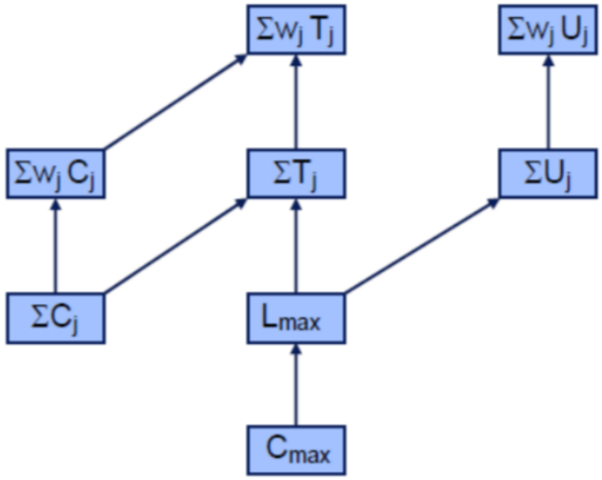
Notation	Definition/Explanation	Notes
r_j	Release dates given	
$prmp$	`Preemption` allowed (fractional processing)	Linear programming, $\therefore$ way easier (opposed to IP w/o $prmp$)
s_{jk}	`Sequence dependent setup times`	Setup times between processing jobs j and k; s_{ijk} has i refer to machine
$prec$	`Precedence constraints`	Usually represented with a directed acyclical graph (DAG)

Note: Due dates are reflected in objective function

3.4.5 Objective Criteria (γ)

Notation/Definition	Explanation	Notes
C_j	`Completion time` of job j	
$L_j := C_j - d_j$	`Lateness` of job j	$+$, - or 0
$T_j := max(C_j, 0)$	`Tardiness` of job j	$+$ or 0 only
$U_j = \begin{cases} 1, & \text{if } C_j > d_j \\ 0, & \text{o.w.} \end{cases}$	`Unit penalty` of job b being late	

3.4.5.1 Hierarchical Representation



E.g. L_{max} solver can be used for C_{max} by setting all $d_j = 0$

3.4.6 Terms

Term	Definition
recirculation	A machine is visited multiple times by the same job
makespan	Time of completion of last job
unforced idleness	Machine not processing anything; Jobs are availbale to be processed by it
forced idleness	Machine not processing anything; No jobs available to be processed by it
non-delay	No unforced idleness
regular objective function	Nondecreasing function when new input of $C' \geq C$ (as in, $C'_1 > C_1, \ldots, C'_n > C_n$
sequence	Of a specific machine. Refers to job order
schedule	Particular allocation of jobs to machines at times
scheduling policy	In stochastic settings, a prescription of appropriate action for a given state

3.4.6.1 Forced Idleness Forced idleness can occur due to:

- precedence relationships
- release dates

3.4.6.2 Irregular functions

- Penalizing earliness as well as lateness
- Negative weights

3.4.6.3 Active Schedule Prereqs:

- Feasible
- Nonpreemptive

Meaning: Not possible to construct another schedule where ≥ 1 operation finishes earlier and no operation finishes later

Basically, no fillable holes in schedule

3.5 Theorems

3.5.1 Theorem 1

For: $\alpha|prmp, \beta|\gamma$

Additional Constraints: γ is regular

Claim: There exists a `non-delay` schedule which is optimal

3.5.1.1 Proof Sketch

3.5.1.1.1 Counter Assume that *all* optimal schedules are *not* non-delay.

3.5.1.1.2 Procedure Definition Let S^1 be such an optimal schedule

Since there is unforced idleness, I can obtain another schedule S^2 by removing a complete segment of the unforced idleness due to preemption from S^1, by filling that idleness with ≥ 1 job

I can thus obtain another schedule S^2 such that $C_j^{S_1} \geq C_j^{S_2}, \forall j$, and where some $\mathrm{C_j^{S_1}} > \mathrm{C_j^{S_2}}$

3.5.1.1.3 Proof of Optimality Due to regularity, this means that $\gamma(C_J^{S_1}) \geq \gamma(C_J^{S_2})$. This implies S^2 is also optimal

3.5.1.1.4 Iteration and Limit of Iteration From this schedule, I can create another schedule S^3, and from that S^4 and so on until I hit a non-delay schedule (the limit of the iterative performance of this operation).

3.5.1.1.5 Conclusion This non-delay schedule is therefore optimal.

3.5.2 Theorem 2

For: $J_m||\gamma$

Additional Constraints: γ is regular

Claim: There exists an `active` schedule which is also optimal

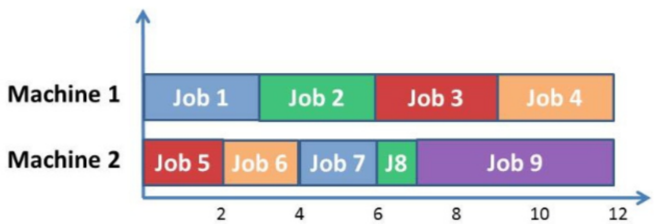
3.6 Anomalies in non-preemptive non-delay schedules

E.g. decreasing processing time may make objective function worse (since you can't delay)

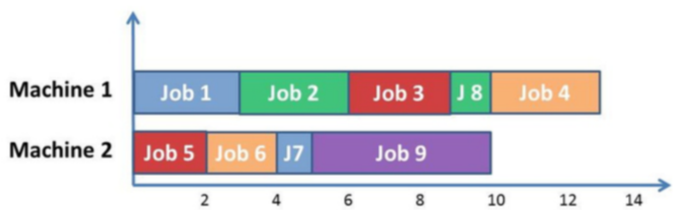
3.6.1 Example

3.6.1.1 Initial

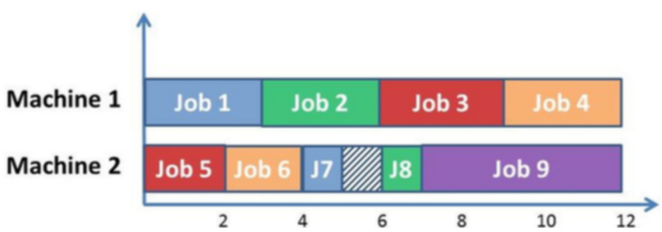
j	1	2	3	4	5	6	7	8	9
p_j	3	3	3	3	2	2	2	1	5
prec_j	-	1	2	3,8	-	5	6	2	7



3.6.1.2 Assuming Job 7 is shorter



3.6.1.3 (Invalid) Better Solution



Invalid due to delay

4 Proving Techniques

4.1 Induction

1. Prove base case (e.g. show P(0) holds)
2. Prove inductive step for general case (e.g. show if P(n) holds then P(n+1) holds)

4.2 Enumeration/Construction/Direct (brute force)

Using definition, show it's always true

4.3 Contradiction (very useful!)

1. Assume counter claim (e.g. theorem says $P \implies Q$, then assume $P \implies \neg Q$)
2. Show contradiction in assumption

4.4 Contra-positive

Show $A \implies B$ due to $\neg B \implies \neg A$

5 TODO To Revise from Optimization

5.1 TODO Simplex

5.2 TODO Dijkstra

5.3 TODO Branch and Bound

6 AMPL

- A Mathematical Programming Language (AMPL)
- Allows for decoupling of the following elements:
  - mathematical programming model
  - data
  - solver