9th Max-Planck Advanced Course on the Foundations of Computer Science (ADFOCS)

Primal-Dual Algorithms for Online Optimization: Lecture 2

Seffi Naor

Computer Science Dept.

Technion

Haifa, Israel

Contents

- Packing problems
  - Routing
  - Load balancing
- General covering/packing results
- More applications

Online Virtual Circuit Routing

Network graph $G=(V, E)$
capacity function $u: E \rightarrow Z^+$

Requests: $r_i = (s_i, t_i)$

- **Problem:** Connect s_i to t_i by a path, or reject the request.
- Reserve one unit of bandwidth along the path.
- **No re-routing is allowed**.
- **Load:** ratio between reserved edge bandwidth and edge capacity.
- **Goal:** Maximize the total throughput.

Routing – Linear Program

$y(r_i, p)$ = Amount of bandwidth allocated for r_i on path p

$P(r_i)$ - Available paths to serve request r_i

$$\max \sum_{r_i} \sum_{p \in P(r_i)} y(r_i, p)$$

s.t:

For each r_i: $\displaystyle\sum_{p \in P(r_i)} y(r_i, p) \le 1$

For each edge e: $\displaystyle\sum_{r_i} \sum_{p \in P(r_i) | e \in p} y(r_i, p) \le u(e)$

Routing – Linear Program

P: Primal Covering

$$\min \sum_{e \in E} u(e)x(e) + \sum_{r_i} z(r_i)$$

$$\forall r_i, p \in P(r_i):$$

$$\sum_{e \in p} \mathrm{x(e)} + z(r_i) \geq 1$$

D: Dual Packing

$$\max \sum_{r_i} \sum_{p \in P(r_i)} y(r_i, p)$$

$$\forall r_i \quad \sum_{p \in P(r_i)} y(r_i, p) \leq 1$$

$$\forall e: \quad \sum_{r_i} \sum_{p \in P(r_i) | e \in p} y(r_i, p) \leq u(e)$$

Online setting:

- **Dual:** new columns arrive one by one.
- **Requirement:** each dual constraint is satisfied.
- **Monotonicity:** variables can only be increased.

Routing – Algorithm 1

P: Primal Covering

$$\min \sum_{e\in E} u(e)x(e) + \sum_{r_i} z(r_i)$$

$$\forall r_i, p \in P(r_i):$$

$$\sum_{e\in p} \mathrm{x(e)} + z(r_i) \geq 1$$

D: Dual Packing

$$\max \sum_{r_i} \sum_{p\in P(r_i)} y(r_i, p)$$

$$\forall r_i \quad \sum_{p\in P(r_i)} y(r_i, p) \leq 1$$

$$\forall e: \quad \sum_{r_i} \sum_{p\in P(r_i) | e\in p} y(r_i, p) \leq u(e)$$

Initially x(e)← 0

When new request arrives, if $\exists p \in P(r_i), \sum_{e\in p} \mathrm{x(e)} < 1$:

- $z(r_i) \leftarrow 1$
- $\forall e \in p: \quad x(e) \leftarrow x(e)\left(1+\frac{1}{u(e)}\right) + \frac{1}{n\cdot u(e)}$
- $y(r_i,p) \leftarrow 1$

Analysis of Algorithm 1

Proof of competitive factor:

1. Primal solution is feasible.
2. In each iteration, $\Delta P \leq 3\Delta D$.
3. Dual is (almost) feasible.

Conclusions: We will see later.

Initially $x(e) \leftarrow 0$

When new request arrives, if $\exists p \in P(r_i), \sum_{e \in p} x(e) < 1$:

- $z(r_i) \leftarrow 1$
- $\forall e \in p: \quad x(e) \leftarrow x(e)\left(1+\frac{1}{u(e)}\right)+\frac{1}{n \cdot u(e)}$
- $y(r_i,p) \leftarrow 1$

Analysis of Algorithm 1

1. Primal solution is feasible.

If $\forall p \in P(r_i), \sum_{e \in p} x(e) \geq 1$: the solution is feasible.

Otherwise: we update z(r_i) ← 1

Initially x(e) ← 0

When new request arrives, if $\exists p \in P(r_i), \sum_{e \in p} x(e) < 1$:

- z(r_i) ← 1
- $\forall e \in p: \quad x(e) \leftarrow x(e)\left(1 + \frac{1}{u(e)}\right) + \frac{1}{n \cdot u(e)}$
- y(r_i,p) ← 1

Analysis of Algorithm 1

2. In each iteration: ΔP ≤ 3ΔD.

If $\forall p \in P(r_i): \sum_{\mathrm{e} \in \mathrm{p}} \mathrm{x(e)} \geq 1$ ΔP = ΔD=0

Otherwise:

ΔD=1

$$\Delta P = \sum_{e \in p} u(e) \Delta x(e) + z(r_i) \Rightarrow = \sum_{e \in p} u(e) \left(\frac{x(e)}{u(e)} + \frac{1}{n \cdot u(e)} \right) + 1 \leq 3$$

Initially x(e)← 0

When new request arrives, if $\exists p \in P(r_i), \sum_{\mathrm{e} \in \mathrm{p}} \mathrm{x(e)} < 1:$

- $z(r_i) \leftarrow 1$
- $\forall e \in p: \quad x(e) \leftarrow x(e) \left(1 + \frac{1}{u(e)} \right) + \frac{1}{n \cdot u(e)}$
- $y(r_i, p) \leftarrow 1$

Analysis of Algorithm 1

3. Dual is (almost) feasible.

We prove:

- For each e, after routing u(e)O(log n) on e, x(e)≥1

  x(e) is a sum of a geometric sequence

  $x(e)_1 = 1/(nu(e))$, $q = 1+1/u(e)$

➔ After u(e)O(log n) requests:

$$x(e) = \frac{1}{n \cdot u(e)} \cdot \frac{\left(1+\frac{1}{u(e)}\right)^{u(e)O(\log n)} - 1}{\left(1+\frac{1}{u(e)}\right) - 1} = \frac{\left(1+\frac{1}{u(e)}\right)^{u(e)O(\log n)} - 1}{n} \geq 1$$

Conclusions: Algorithm 1

- The algorithm is 3-competitive, since $\Delta P \leq 3\Delta D$
- Edge capacities are violated by at most a factor of $O(\log n)$, since the dual is "almost" feasible.

Routing – Algorithm 2

P: Primal Covering

$$\min \sum_{e \in E} u(e)x(e) + \sum_{r_i} z(r_i)$$

$$\forall r_i, p \in P(r_i):$$

$$\sum_{e \in p} x(e) + z(r_i) \geq 1$$

D: Dual Packing

$$\max \sum_{r_i} \sum_{p \in P(r_i)} y(r_i, p)$$

$$\forall r_i \quad \sum_{p \in P(r_i)} y(r_i, p) \leq 1$$

$$\forall e: \quad \sum_{r_i} \sum_{p \in P(r_i) | e \in p} y(r_i, p) \leq u(e)$$

Initially: $\forall e, x(e) \leftarrow 0$

For new request r_i , if $\exists p \in P(r_i), \sum_{e \in p} x(e) < 1$:

- $z(r_i) \leftarrow 1$
- $\forall e \in p: \; x(e) \leftarrow x(e) \cdot \exp\left(\frac{\ln(1+n)}{u(e)}\right) + \frac{1}{n}\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right) - 1\right]$
- $y(r_i, p) \leftarrow 1$

Analysis of Algorithm 2

Proof of competitive factor:

1. Primal solution is feasible.
2. In each iteration, $\Delta P \approx O(\log n)\Delta D$.
3. Dual is feasible.

Initially: $\forall e, x(e) \leftarrow 0$

For new request r_i , if $\exists p \in P(r_i), \ \sum_{e \in p} x(e) < 1$:

- $z(r_i) \leftarrow 1$
- $\forall e \in p: \ x(e) \leftarrow x(e) \cdot \exp\left(\frac{\ln(1+n)}{u(e)}\right) + \frac{1}{n}\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right) - 1\right]$
- $y(r_i, p) \leftarrow 1$

Analysis of Algorithm 2

1. Primal solution is feasible.

   If $\forall p \in P(r_i), \sum_{e \in p} x(e) \geq 1$: the solution is feasible.

   Otherwise: we update $z(r_i) \leftarrow 1$

Initially: $\forall e, x(e) \leftarrow 0$

For new request r_i , if $\exists p \in P(r_i), \ \sum_{e \in p} x(e) < 1$:

- $z(r_i) \leftarrow 1$
- $\forall e \in p: \ x(e) \leftarrow x(e) \cdot \exp\left(\frac{\ln(1+n)}{u(e)}\right) + \frac{1}{n}\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right) - 1\right]$
- $y(r_i, p) \leftarrow 1$

Analysis of Algorithm 2

2. Ratio between ΔP and ΔD: If $\forall p \in P(r_i): \sum_{e \in p} x(e) \geq 1$, ΔP = ΔD=0

Otherwise: ΔD=1 and

ΔP =

$$1 + \sum_{e \in p} u(e) \left(x(e) \left[\exp\left(\frac{\ln(1+n)}{u(e)} \right) - 1 \right] + \frac{1}{n} \left[\exp\left(\frac{\ln(1+n)}{u(e)} \right) - 1 \right] \right)$$

Initially: $\forall e, x(e) \leftarrow 0$

For new request r_i , if $\exists p \in P(r_i), \ \sum_{e \in p} x(e) < 1$:

- $z(r_i) \leftarrow 1$
- $\forall e \in p: \ x(e) \leftarrow x(e) \cdot \exp\left(\frac{\ln(1+n)}{u(e)}\right) + \frac{1}{n}\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right) - 1\right]$
- $y(r_i, p) \leftarrow 1$

Analysis of Algorithm 2

$$\left(u(e)\cdot\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right)-1\right]\right)$$ - monotonically decreasing

Therefore, ΔP is at most:

$$1+\sum_{e\in P}u(e)\left(x(e)\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right)-1\right]+\frac{1}{n}\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right)-1\right]\right)$$

$$\leq\ 2\left(u(\min)\cdot\left[\exp\left(\frac{\ln(1+n)}{u(\min)}\right)-1\right]\right)+1$$

since: $z(r_i)=1 \text{ and } \sum_{e\in p}x(e)\leq 1$

Thus, ΔP/ ΔD $\leq\ 2\left(u(\min)\cdot\left[\exp\left(\frac{\ln(1+n)}{u(\min)}\right)-1\right]\right)+1$

Analysis of Algorithm 2

3. Dual is feasible. We prove:
   – For each e, after routing u(e) requests, x(e)≥1

x(e) is a sum of a geometric sequence

$$(x(e))_1 = \frac{1}{n}\left[\exp\left(\frac{\ln(1+n)}{u(e)}\right) - 1\right] \text{ and } q = \exp\left(\frac{\ln(1+n)}{u(e)}\right)$$

➔ After u(e) requests:

$$\begin{aligned} x(e) &= \frac{1}{n}\cdot\left(\exp\left(\frac{\ln(1+n)}{u(e)}\right) - 1\right)\cdot\frac{\exp\left(\frac{u(e)\ln(1+n)}{u(e)}\right) - 1}{\exp\left(\frac{\ln(1+n)}{u(e)}\right) - 1} \\ &= \frac{1}{n}\cdot(1+n-1) \geq 1. \end{aligned}$$

✓

Conclusions: Algorithm 2

- $O\left(u(\min) \cdot \left[\exp\left(\frac{\ln(1+n)}{u(\min)}\right) - 1\right]\right)$ – competitive
- It does not violate capacity constraints
- If $u(\min) \geq \log n$ then,

$$2\left(u(\min) \cdot \left[\exp\left(\frac{\ln(1+n)}{u(\min)}\right) - 1\right]\right) + 1 = O(\log n)$$

- This result was obtained by [AAP, 1993]

Further Results: Routing

We saw a simple algorithm which is:

- **3-competitive** and violates capacities by O(log n) factor.

Can be improved [Buchbinder, N., FOCS06] to:

- **1-competitive** and violates capacities by O(log n) factor.

Non Trivial.

Main ideas:

- Combination of ideas drawn from casting of previous routing algorithms within the primal-dual approach.
- Decomposition of the graph.
- Maintaining **several primal solutions** which are used to bound the dual solution, and for the routing decisions.

Further Results: Routing

Applications [Buchbinder, N, FOCS 06]:

- Can be used as “black box” for many objective functions and in many routing models:
  - **Previous Settings** [AAP93,APPFW94].
  - **Maximizing throughput.**
  - **Minimizing load.**
  - **Achieving better global fairness results (Coordinate competitiveness).**

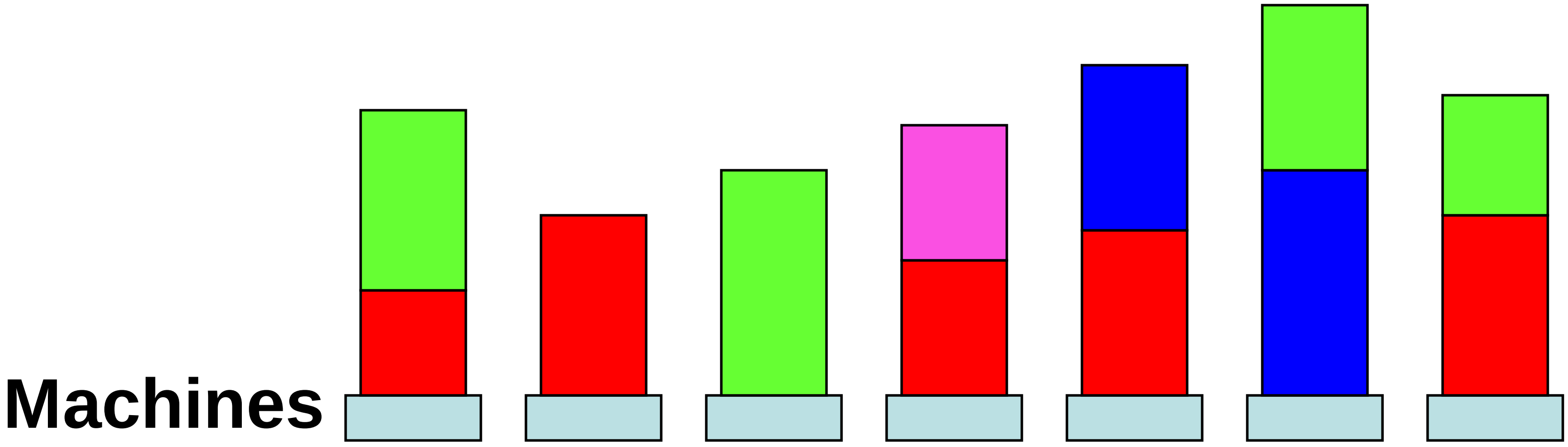
Scheduling and Load Balancing

- Set of m machines
- Set of jobs
- Assigning a job to a machine incurs a load

Motivation and Objective

- Parallel processing of jobs on machines
- Assignments of packets to communication lines
- Distributing web cache files on web servers

Objective: minimize maximum load - makespan



Machine Scheduling Models

Identical machines:

- A job can be assigned to any machine, incurring the same load

Restricted assignment:

- A job can be assigned to only a subset of the machines
- The load of a job on all allowed machines is the same

Unrelated machines: **[our focus]**

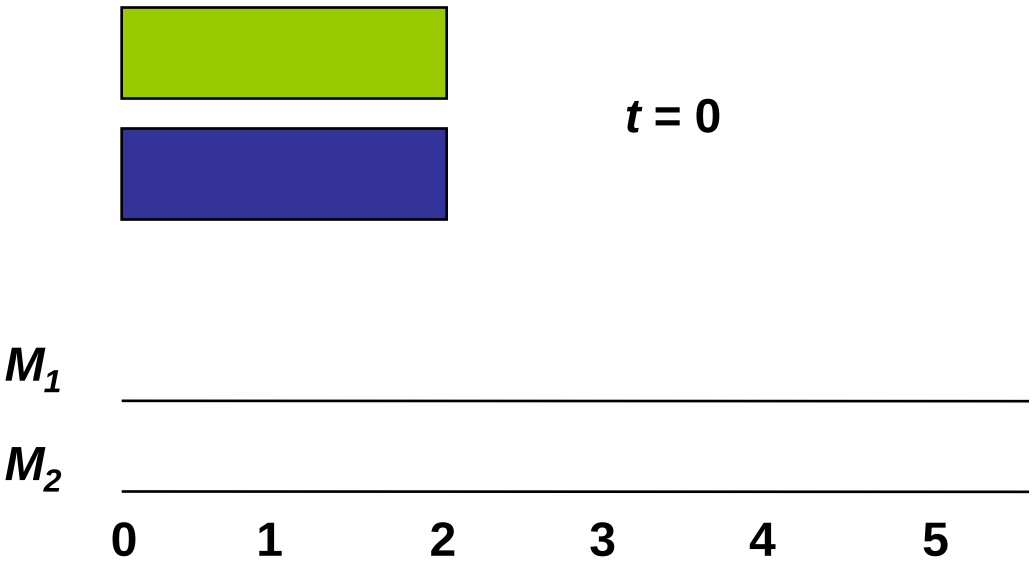
- Job i on machine j has load p(i,j)

Online Model

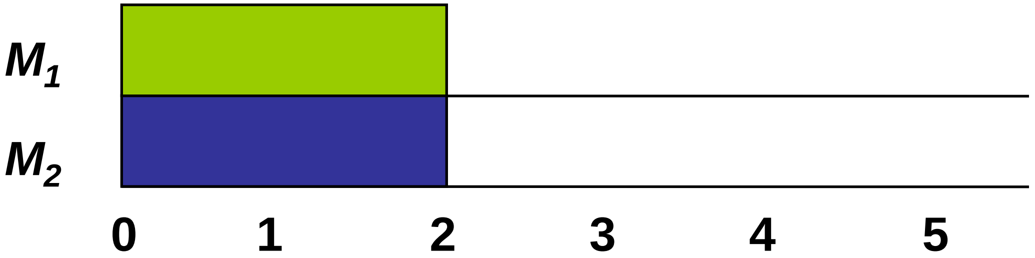
Online setting:

- Jobs arrive one-by-one
- Upon arrival of each job:
  - reveals its load function
  - needs to be assigned to a machine
- Assignments of jobs to machines are irreversible

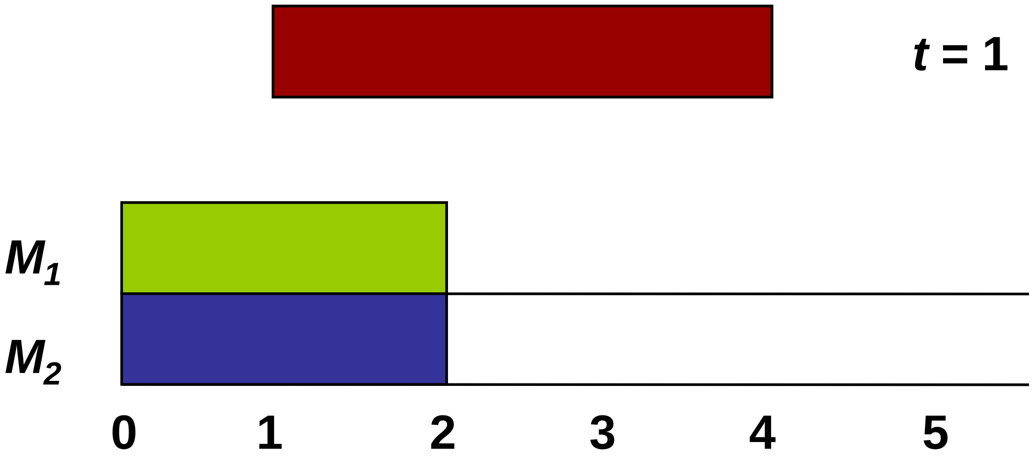
Example



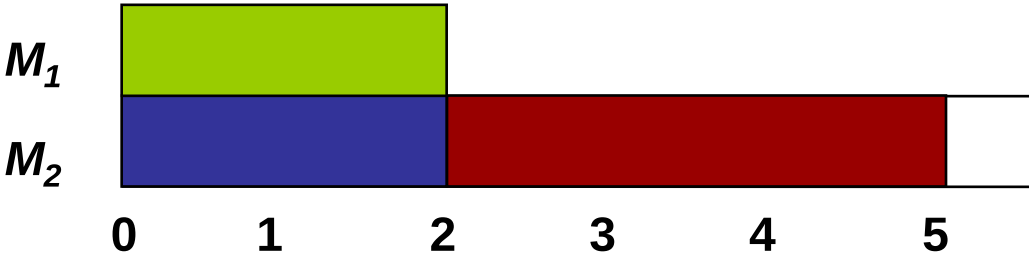
Example



Example

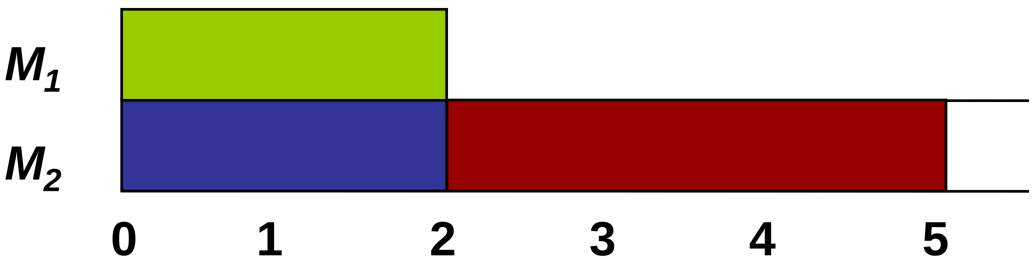


Example

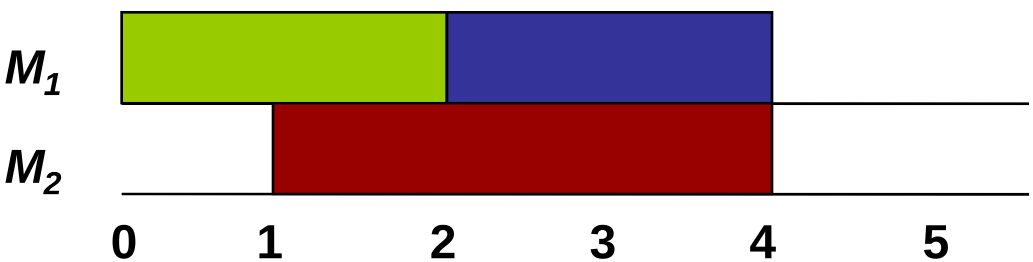


On-line solution

Example



On-line solution



Optimal solution

Our Model

Unrelated machines:

- Job i on machine j has load p(i,j)

Linear program:

- we want to write a maximization program
- we assume that OPT's max load α is known
- obtained by "doubling":
  - it guarantees α ≤ 2·(OPT's max load)

Doubling

- Initially: $\alpha \leftarrow$ minimum load (known)
- Our online algorithm keeps the invariant:
  - either its max load $\leq \alpha \cdot$(competitive ratio)
  - or it generates a certificate that OPT $> \alpha$ (“failure”)
- In case of **failure**:
  - $\alpha \leftarrow 2 \cdot \alpha$ ($\alpha \leq 2 \cdot$OPT is maintained)
  - “forget” about previous assignments
  - assignments for different α-s are geometric:
    [$\alpha \cdot$(competitive ratio) + 2 $\alpha \cdot$(competitive ratio) + 4 $\alpha \cdot$(competitive ratio) + …]
  - loss incurred is at most a factor of 4

Setting up the Linear Program (2)

- Normalized load of job j on machine i: $\tilde{p}(i,j) = \frac{p(i,j)}{\alpha}$
- Upon arrival of job j:
  - machine i is eligible if $\tilde{p}(i,j) \leq 1$
  - no such machine exists: announce failure!
  - clearly, OPT also cannot schedule with load ≤ α

Linear Program: fixed α

Primal		Dual	
min	$\sum_j x(j) + \sum_i z(i)$	max	$\sum_i \sum_{j \in E(i)} y(i,j)$
subject to:		subject to:	
$\forall i, j \in E(i)$:	$\tilde{p}(i,j)x(j) + z(i) \geq 1$	$\forall i$:	$\sum_{j \in E(i)} y(i,j) \leq 1$
		$\forall j$:	$\sum_i \tilde{p}(i,j) y(i,j) \leq 1$

y(i,j) – indicator for scheduling job i on machine j

Objective: maximize number of jobs scheduled

- If max load is correctly guessed, then all jobs can be scheduled!

Load Balancing Algorithm: fixed α

Initially: $x(j) \leftarrow \frac{1}{2m}$.

Upon arrival of job i:

1. If there is no machine j such that $\tilde{p}(i,j) \leq 1$, or there exists a machine with $x(j) > 1$, return "failure". Otherwise:
   (a) Let $\ell \in E(i)$ be a machine minimizing $\tilde{p}(i,\ell)x(\ell)$.
   (b) Assign job i to machine ℓ: $y(i,\ell) \leftarrow 1$.
   (c) $z(i) \leftarrow 1 - \tilde{p}(i,\ell)x(\ell)$.
   (d) $x(\ell) \leftarrow x(\ell)(1 + \frac{\tilde{p}(i,\ell)}{2})$.

Primal		Dual	
min	$\sum_j x(j) + \sum_i z(i)$	max	$\sum_i \sum_{j \in E(i)} y(i,j)$
subject to:		subject to:	
$\forall i, j \in E(i)$:	$\tilde{p}(i,j)x(j) + z(i) \geq 1$	$\forall i$:	$\sum_{j \in E(i)} y(i,j) \leq 1$
		$\forall j$:	$\sum_i \tilde{p}(i,j)y(i,j) \leq 1$

Analysis of Load Balancing Algorithm

We show:

- Load of assigned jobs on each machine is $O(\alpha \cdot \log m)$

- If algorithm returns **failure**: then there exists a primal solution of value < N (# of jobs) – a certificate that OPT> α

- Else: all jobs are scheduled with load $O(\alpha \cdot \log m)$

Bounding the Load on the Machines

- Since $\tilde{p}(i,j) \leq 1$, $x(j) \leq 3/2$
- Hence:

$$\begin{aligned} 3/2 \geq x(j) \geq \frac{1}{2m} \cdot \prod_{i \in j} \left(1 + \frac{\tilde{p}(i,j)}{2}\right) &\geq \frac{1}{2m} \cdot \prod_{i \in j} \left(\frac{4}{3}\right)^{\tilde{p}(i,j)} \\ &= \frac{1}{2m} \cdot \exp\left(\ln\left(\frac{4}{3}\right) \cdot \sum_{i \in j} \tilde{p}(i,j)\right) \end{aligned}$$

- Simplifying:

$$\sum_{i \in j} \tilde{p}(i,j) \leq \frac{\ln(3m)}{\ln\left(\frac{4}{3}\right)} = O(\log m)$$

- Holds also in case of failure

The Primal Solution

Why is the primal solution feasible:

- consider constraint $\tilde{p}(i,j)x(j) + z(i) \geq 1$
- for each job i, $z(i) \leftarrow 1 - p(i,\ell)x(\ell)$, where ℓ minimizes $\tilde{p}(i,\ell)x(\ell)$
- thus, all primal constraints related to i are satisfied
- since $x(i)$ is increasing, constraints remain feasible

When assigning job i to machine ℓ: $(P = \sum_j x(j) + \sum_i z(i)\ )$

- $\Delta P = 1 - p(i,\ell)x(\ell) + \frac{p(i,\ell)x(\ell)}{2} = 1 - \frac{p(i,\ell)x(\ell)}{2}$
- $\Delta x(\ell) = \frac{p(i,\ell)x(\ell)}{2}$

The Primal Solution

- $\Delta P = 1 - \Delta x(\ell)$
- N - number of jobs
- $x(j)_{\text{init}} = \frac{1}{2m}$

Thus,

$$
\begin{aligned}
P &= \sum_{j=1}^{m} x(j)_{\text{init}} + N - \sum_{j=1}^{m} \left(x(j) - x(j)_{\text{init}}\right) \\
&= 2 \cdot \sum_{j=1}^{m} x(j)_{\text{init}} + N - \sum_{j=1}^{m} x(j) = 1 + N - \sum_{j=1}^{m} x(j)
\end{aligned}
$$

If $\exists\ x(j) > 1$, then $P < N$, failure! We have a certificate that OPT$> \alpha$

Online Primal-Dual Approach: Summary

- Can the **offline** problem be cast as a **linear covering/packing program**?
- Can the online process be described as:
  - **New rows appearing in a covering LP?**
  - **New columns appearing in a packing LP?**

Yes ??

- Upon arrival of a new request:
  - Update primal variables in a **multiplicative way**.
  - Update dual variables in an **additive way**.

Online Primal Dual Approach

Next Prove:

1. Primal solution is **feasible** (or nearly feasible).
2. In each round, **$\Delta P \leq c\ \Delta D$**.
3. Dual is **feasible** (or **nearly feasible**).

Got a **fractional** solution, but need an **integral** solution ??

- Randomized rounding techniques might work.
- Sometimes, even derandomization (e.g., method of conditional probabilities) can be applied online!

Online Primal-Dual Approach

Advantages:

1. **Generic** ideas and algorithms applicable to many online problems.
2. **Linear Program** helps detecting the difficulties of the online problem.
3. **General recipe** for the design and analysis of online algorithms.
4. No **potential function** appearing **“out of nowhere”**.
5. Competitiveness with respect to a **fractional optimal solution.**

General Covering/Packing Results

What can you expect to get?

- For a {0,1} covering/packing matrix:
  - **Competitive ratio O(log D)** [BN05]

  (D – max number of non-zero entries in a constraint).

Remarks:

- Fractional solutions.
- Number of constraints/variables can be exponential.
- There can be a tradeoff between the competitive ratio and the factor by which constraints are violated.

General Covering/Packing Results

- For a **general covering/packing** matrix [BN05] :

Covering:

- Competitive ratio O(log n)

  (n – number of variables).

Packing:

- Competitive ratio O(log n + log [a(max)/a(min)])

  a(max), a(min) – maximum/minimum non-zero entry

Remarks:

- Results are tight.

Further Results via P-D Approach

Covering Online Problems (Minimization):

- **Dynamic TCP Acknowledgement**
- **Parking Permit Problem** [Meyerson 05]
- **Online Graph Covering Problems** [AAABN04]:
  - Non-metric facility location
  - Generalized connectivity: pairs arrive online
  - Group Steiner: groups arrive online
  - Online multi-cut: (s,t)--pairs arrive online