Distributed Optimization – Scheduling Problem

Scheduling Problem

- Set of time slots and set of agents
- Each agent requires some number of time slots and has a deadline
- Shared resource
- Each agent has value for completion of task before deadline

Scheduling Problem

- N -- set of n agents
- X -- set of m discrete, consecutive time slots
- $q = (q_1, q_2, .., q_m)$ -- reserve price vector
- $v = (v_1, v_2, ..., v_n)$ -- valuation functions
  - $v_i(F_i) = w_i$ *if* F_i *includes* λ_i *hours before* d_i, *0 otherwise*

Scheduling Problem

- Solution vector $F = (F_\varnothing, F_1, \ldots, F_n)$, where F_i is the set of time slots assigned to agent i. $F_\varnothing$ is the time slots that are not assigned
- Value of solution:

$$V(F) = \sum_{j | x_j \in F_\emptyset} q_j + \sum_{i \in N} v_i(F_i).$$

Example

- Scheduling jobs on a processor. Eight one-hour time slots from 9 am to 5 pm.
- Reserve price: \$3 per hour
- Four jobs, each with its own length, deadline, and worth

job	**length** (λ)	**deadline** (d)	**worth** (w)
1	2 hours	1:00 P.M.	\$10.00
2	2 hours	12:00 P.M.	\$16.00
3	1 hours	12:00 P.M.	\$6.00
4	4 hours	5:00 P.M.	\$14.50

Example

job	length (λ)	deadline (d)	worth (w)
1	2 hours	1:00 P.M.	$10.00
2	2 hours	12:00 P.M.	$16.00
3	1 hours	12:00 P.M.	$6.00
4	4 hours	5:00 P.M.	$14.50

time slot	agent
9:00 A.M.	2
10:00 A.M.	2
11:00 A.M.	1
12:00 P.M.	1
13:00 P.M.	4
14:00 P.M.	4
15:00 P.M.	4
16:00 P.M.	4

Scheduling Problem

- NP-complete
- Integer program:

$$
\begin{aligned}
\text{maximize} \quad & \sum_{S \subseteq X, i \in N} v_i(S) x_{i,S} \\
\text{subject to} \quad & \sum_{S \subseteq X} x_{i,S} \leq 1 && \forall i \in N \\
& \sum_{S \subseteq X : j \in S, i \in N} x_{i,S} \leq 1 && \forall j \in X \\
& x_{i,S} \in \{0, 1\} && \forall S \subseteq X, i \in N
\end{aligned}
$$

Competitive Equilibrium

- Generalize notion of competitive equilibrium to the scheduling problem

Definition 2.3.11 (Competitive equilibrium, generalized form) *Given a scheduling problem, a solution F is in* competitive equilibrium *at prices p if and only if*

- *For all $i \in N$ it is the case that $F_i = \arg\max_{T \subseteq X}(v_i(T) - \sum_{j|x_j \in T} p_j)$ (the set of time slots allocated to agent i maximizes his surplus at prices p);*
- *For all j such that $x_j \in F_\emptyset$ it is the case that $p_j = q_j$ (the price of all unallocated time slots is the reserve price); and*
- *For all j such that $x_j \notin F_\emptyset$ it is the case that $p_j \geq q_j$ (the price of all allocated time slots is greater than the reserve price).*

Competitive Equilibrium

- Example

job	length (λ)	deadline (d)	worth (w)
1	2 hours	1:00 P.M.	$10.00
2	2 hours	12:00 P.M.	$16.00
3	1 hours	12:00 P.M.	$6.00
4	4 hours	5:00 P.M.	$14.50

time slot	agent	price
9:00 A.M.	2	$6.25
10:00 A.M.	2	$6.25
11:00 A.M.	1	$6.25
12:00 P.M.	1	$3.25
13:00 P.M.	4	$3.25
14:00 P.M.	4	$3.25
15:00 P.M.	4	$3.25
16:00 P.M.	4	$3.25

Competitive Equilibrium

- Theorem: If a solution F to a scheduling problem C is in equilibrium at prices p, then F is optimal.

Competitive Equilibrium

$$
\begin{aligned}
V(F) &= \sum_{j|x_j \in F_\emptyset} q_j + \sum_{i \in N} v_i(F_i) \\
&= \sum_{j|x_j \in F_\emptyset} p_j + \sum_{i \in N} v_i(F_i) \\
&= \sum_{j|x_j \in X} p_j + \sum_{i \in N} \left[v_i(F_i) - \sum_{j|x_j \in F_i} p_j \right] \\
&\geq \sum_{j|x_j \in X} p_j + \sum_{i \in N} \left[v_i(F'_i) - \sum_{j|x_j \in F'_i} p_j \right] = V(F')
\end{aligned}
$$

Competitive Equilibrium

- Competitive equilibrium need not exist
- Consider processor with two slots, 9 am and 10 am, reserve price $3

job	length (λ)	deadline (d)	worth (w)
1	2 hours	11:00 A.M.	$10.00
2	1 hour	11:00 A.M.	$6.00

Competitive Equilibrium

- Theorem: A scheduling problem has competitive equilibrium solution iff the LP relaxation has an integer solution.

Auction algorithm

foreach *slot* x_j **do**
    $b_j \leftarrow q_j$
    // Set the initial bids to be the reserve price
foreach *agent* i **do**
    $F_i \leftarrow \emptyset$
repeat
    foreach *agent* $i = 1$ *to* n **do**
        foreach *slot* x_j **do**
            if $x_j \in F_i$ **then**
                $p_j \leftarrow b_j$
            else
                $p_j \leftarrow b_j + \epsilon$
            // Agents assume that they will get slots they are currently the high bidder on at that price, while they must increment the bid by ϵ to get any other slot.
        $S^* \leftarrow \arg\max_{S \subseteq X | S \supseteq F_i} (v_i(S) - \sum_{j \in S} p_j)$
        // Find the best subset of slots, given your current outstanding bids
        // Agent i becomes the high bidder for all slots in $S^* \setminus F_i$.
        foreach *slot* $x_j \in S^* \setminus F_i$ **do**
            $b_j \leftarrow b_j + \epsilon$
            if *there exists an agent* $k \neq i$ *such that* $x_j \in F_k$ **then**
                set $F_k \leftarrow F_k \setminus \{x_j\}$
            // Update the bidding price and current allocations of the other bidders.
        $F_i \leftarrow S^*$
until F *does not change*

Auction algorithm

job	length (λ)	deadline (d)	worth (w)
1	2 hours	1:00 P.M.	\$10.00
2	2 hours	12:00 P.M.	\$16.00
3	1 hours	12:00 P.M.	\$6.00
4	4 hours	5:00 P.M.	\$14.50

round	bidder	slots bid on	$\mathbf{F} = (\mathbf{F_1}, \mathbf{F_2}, \mathbf{F_3}, \mathbf{F_4})$	b
0	1	(9,10)	$(\{9, 10\}, \{\emptyset\}, \{\emptyset\}, \{\emptyset\})$	(3.25,3.25,3,3,3,3,3,3)
1	2	(10,11)	$(\{9\}, \{10, 11\}, \{\emptyset\}, \{\emptyset\})$	(3.25,3.5,3.25,3,3,3,3,3)
2	3	(9)	$(\{\emptyset\}, \{10, 11\}, \{9\}, \{\emptyset\})$	(3.5,3.5,3.25,3,3,3,3,3)
⋮	⋮	⋮	⋮	⋮
24	1	$\emptyset$	$(\{11, 12\}, \{9, 10\}, \{\emptyset\}, \{12, 13, 14, 15\})$	(6.25,6.25,6.25,3.25, 3.25,3.25,3.25,3.25)