Online Scheduling with Interval Conflicts

Dror Rawitz
Tel-Aviv University

Joint work with Magnús M. Halldórsson and Boaz Patt-Shamir
Online Algorithm:

- Input is revealed piece by piece over time
- Algorithm must make irrevocable decisions without access to whole input
Online Algorithms & Competitive Analysis

Online Algorithm:
- Input is revealed piece by piece over time
- Algorithm must make irrevocable decisions without access to whole input

Competitive Analysis:
- Solution is compared to offline OPT
- ALG has competitive ratio $c$ if $\exists \alpha$ s.t.

$$\forall I, \ ALG(I) \geq \frac{OPT(I)}{c} - \alpha$$
Online Algorithms & Competitive Analysis

Online Algorithm:
- Input is revealed piece by piece over time
- Algorithm must make irrevocable decisions without access to whole input

Competitive Analysis:
- Solution is compared to offline OPT
- ALG has competitive ratio $c$ if $\exists \alpha$ s.t.
  \[ \forall I, \quad ALG(I) \geq \frac{OPT(I)}{c} - \alpha \]
- Game against an adversary
  - Knows algorithm
  - Determines the input sequence
  - Can obtain offline optimum
Problem Statement

Input:
- Set $U$ of items, each with an integer identifier
- Collection $C$ of *interval conflicts*
- Each conflict $C \in C$ contains all items within some interval
Problem Statement

□ Input:
- Set $U$ of items, each with an integer identifier
- Collection $C$ of *interval conflicts*
- Each conflict $C \in C$ contains all items within some interval

□ Algorithm:
- Given a conflict, decides which item survives
- All other items in the conflict set are eliminated

□ Goal:
- Maximize number of items that survive all their conflicts
Examples

- Rectangles represent conflicts
- A dot represents an item that survived a conflict

Example 1:
- Solution 1: \{5\}
Examples

- Rectangles represent conflicts
- A dot represents an item that survived a conflict

Example 1:
- Solution 1: \{5\}
- Solution 2: \{1, 3, 6, 8\}
Examples

- Rectangles represent conflicts
- A dot represents an item that survived a conflict

Example 1:
- Solution 1: \{5\}
- Solution 2: \{1, 3, 6, 8\}

Example 2:
- How would “choose leftmost” work? Distributively?
Examples

- Rectangles represent conflicts
- A dot represents an item that survived a conflict

Example 1:
- Solution 1: \{5\}
- Solution 2: \{1, 3, 6, 8\}

Example 2:
- How would “choose leftmost” work? Distributively?
Related Work: Multipart Tasks

Online Set Packing: [Emek, Halldórsson, Mansour, Patt-Shamir, Radhakrishnan, R '10]

- Conflicts are not required to be intervals
- \( k = \max \) number of conflicts per item \((\text{max set size})\)
- \( \sigma = \max \) conflict size \((\text{max element degree})\)
- \( \Omega(\sigma^{k-1}) \) deterministic lower bound
- \( \tilde{\Theta}(k\sqrt{\sigma}) \) randomized competitive ratio
Related Work: Multipart Tasks

- **Online Set Packing:** [Emek, Halldórsson, Mansour, Patt-Shamir, Radhakrishnan, R ’10]
  - Conflicts are not required to be intervals
  - $k = \max\ \text{number of conflicts per item}$ (max set size)
  - $\sigma = \max\ \text{conflict size}$ (max element degree)
  - $\Omega(\sigma^{k-1})$ deterministic lower bound
  - $\tilde{\Theta}(k\sqrt{\sigma})$ randomized competitive ratio

- **Packets with Dependencies:** [Kesselman, Patt-Shamir, Scalosub 09]
  - Data items composed of $k$ packets arrive at a FIFO buffer
  - No competitive deterministic online algorithm
  - Order respecting arrival sequences:
    - Deterministic lower bound $\Omega(k)$
    - Deterministic upper bound $O(k^2)$
Problem Variants

Centralized vs. Distributed:

□ Sequential model:
  – Conflicts arrive at the same location, sequentially
  – Algorithm resolves them one by one, knowing past conflicts and decisions

□ Oblivious model:
  – Conflicts arrive in different locations, possibly in parallel
  – Algorithm required to resolve conflicts without knowledge of other conflicts
Problem Variants

Centralized vs. Distributed:

□ Sequential model:
  – Conflicts arrive at the same location, sequentially
  – Algorithm resolves them one by one, knowing past conflicts and decisions

□ Oblivious model:
  – Conflicts arrive in different locations, possibly in parallel
  – Algorithm required to resolve conflicts without knowledge of other conflicts

Conflicts Type:

□ Contiguous: \( C = [\min(C), \max(C)] \)

□ General: \( C = U \cap [\min(C), \max(C)] \)
## Our Results

- **Deterministic competitive ratio**

<table>
<thead>
<tr>
<th></th>
<th>sequential</th>
<th>oblivious</th>
</tr>
</thead>
<tbody>
<tr>
<td>contiguous</td>
<td>$\Omega(\log \sigma)$</td>
<td>$O(\log \sigma)$</td>
</tr>
<tr>
<td>general</td>
<td>$O(\log \sigma)$</td>
<td>$\Omega(n)$</td>
</tr>
</tbody>
</table>

- $\sigma = \text{maximum size of a conflict}$
- $n = \text{number of items}$
- Sequential algorithm applies to weighted case
- $\Omega(n)$ applies to $\sigma = 2$
Our Results

- Deterministic competitive ratio

<table>
<thead>
<tr>
<th></th>
<th>sequential</th>
<th>oblivious</th>
</tr>
</thead>
<tbody>
<tr>
<td>contiguous</td>
<td>$\Omega(\log \sigma)$</td>
<td>$O(\log \sigma)$</td>
</tr>
<tr>
<td>general</td>
<td>$O(\log \sigma)$</td>
<td>$\Omega(n)$</td>
</tr>
</tbody>
</table>

- $\sigma =$ maximum size of a conflict
- $n =$ number of items
- Sequential algorithm applies to weighted case
- $\Omega(n)$ applies to $\sigma = 2$

- Additional results:
  - $O(\log(\sigma/b))$-competitive oblivious algorithm for the case where $b$ items may survive each conflict
  - $O(k)$ sequential algorithm
  - 1-competitive algorithm when allowed to accept two items per conflict
Observation:
If all items can be covered by $m$ conflicts, then $\text{OPT} \leq m$
Observation:
If all items can be covered by \( m \) conflicts, then \( \text{OPT} \leq m \)

Definition:
Given an execution, an elimination chain is a sequence of items \( i_0, \ldots, i_m \) s.t. \( i_j \) eliminated \( i_{j-1} \).
Elimination Chains

- **Observation:**
  If all items can be covered by $m$ conflicts, then $\text{OPT} \leq m$

- **Definition:**
  Given an execution, an *elimination chain* is a sequence of items $i_0, \ldots, i_m$ s.t. $i_j$ eliminated $i_{j-1}$

- **Lemma:**
  If all elimination chains of $\text{ALG}$ are shorter than $m$, then $\text{ALG}$ has competitive ratio at most $2m$

**Proof:**
From each chain $\text{ALG}$ gets 1 while $\text{OPT}$ gets at most $2m$
Oblivious Algorithm for Contiguous Instances

Algorithm:

- Priority: \( p(i) \triangleq \max \{ \ell : 2^\ell \text{ divides } i \} \)
- Decision rule: choose task with highest priority
Oblivious Algorithm for Contiguous Instances

- **Algorithm:**
  - **Priority:** \( p(i) \triangleq \max \{ \ell : 2^\ell \text{ divides } i \} \)
  - **Decision rule:** choose task with highest priority

- **Properties:**
  - Well defined: task with highest priority always exists
  - Oblivious
  - No need to know \( \sigma \)
Oblivious Algorithm for Contiguous Instances

- Algorithm:
  - Priority: \( p(i) \triangleq \max \{ \ell : 2^\ell \text{ divides } i \} \)
  - Decision rule: choose task with highest priority

- Properties:
  - Well defined: task with highest priority always exists
  - Oblivious
  - No need to know \( \sigma \)

- Analysis:
  - Any interval contains at most one task with \( p(i) \geq \log \sigma \)
  - Length of any elimination chain is at most \( \log \sigma \)
  - Competitive ratio \( \leq 2 \log \sigma \)
Fix a deterministic oblivious algorithm \( \text{ALG} \)

**Construction:**

- 2-color the edges of \( K_N \):
  - \((v_i, v_j), \) for \( i < j \), is blue if \( i \prec_{\text{ALG}} j \), and otherwise red

\[ \Rightarrow \]

By Ramsey's theorem \( K_N \) contains a monochromatic subgraph of \( n = \Omega(\log N) \) vertices
□ Fix a deterministic oblivious algorithm $\text{ALG}$

□ **Construction:**

- 2-color the edges of $K_N$: $(v_i, v_j)$, for $i < j$, is blue if $i \prec_{\text{ALG}} j$, and otherwise red

$\Rightarrow$ By Ramsey’s theorem $K_N$ contains a monochromatic subgraph of $n = \Omega(\log N)$ vertices

$\Rightarrow$ Increasing/decreasing sequence of $n$ items $i_1, \ldots, i_n$ such that $\text{ALG}$ prefers $i_\ell$ over $i_{\ell-1}$, for any $\ell$

- We introduce the conflicts $\{i_{\ell-1}, i_\ell\}$, for $\ell \in \{2, \ldots, n\}$
Fix a deterministic oblivious algorithm $\text{ALG}$

Construction:

- 2-color the edges of $K_N$: $(v_i, v_j)$, for $i < j$, is blue if $i <_{\text{ALG}} j$, and otherwise red

$\Rightarrow$ By Ramsey's theorem $K_N$ contains a monochromatic subgraph of $n = \Omega(\log N)$ vertices

$\Rightarrow$ Increasing/decreasing sequence of $n$ items $i_1, \ldots, i_n$ such that $\text{ALG}$ prefers $i_\ell$ over $i_{\ell-1}$, for any $\ell$

- We introduce the conflicts $\{i_{\ell-1}, i_\ell\}$, for $\ell \in \{2, \ldots, n\}$

Analysis:

- Only $i_n$ survives the execution of $\text{ALG}$
- $\{i_\ell : \ell \text{ is odd}\}$ is feasible and of size $n/2$

$\Rightarrow$ Competitive ratio is $\Omega(n)$
For each item $i$:
- **Weight class**: $c(i) = \lfloor \log w(i) \rfloor$
- **Left level**: $\text{left}(i)$
- **Right level**: $\text{right}(i)$
Sequential Algorithm

- For each item $i$:
  - Weight class: $c(i) = \lfloor \log w(i) \rfloor$
  - Left level: $\text{left}(i)$
  - Right level: $\text{right}(i)$

- **Algorithm**: Given a conflict
  - Drop all items but leftmost $l$ and rightmost $r$ in highest weight class
  - If $l = r$  $\implies$ $\sqrt{\phantom{0}}$
  - Else
    - If $\text{left}(l) > \text{right}(r)$  $\implies$ $l$ survives; $\text{right}(l) \leftarrow \text{right}(r) + 1$
    - Else  $\implies$ $r$ survives; $\text{left}(r) \leftarrow \text{left}(l) + 1$
Sequential Algorithm

- For each item $i$:
  - Weight class: $c(i) = \lfloor \log w(i) \rfloor$
  - Left level: $\text{left}(i)$
  - Right level: $\text{right}(i)$

- Algorithm: Given a conflict
  - Drop all items but leftmost $l$ and rightmost $r$ in highest weight class
  - If $l = r \implies \checkmark$
  - Else
    - If $\text{left}(l) > \text{right}(r) \implies l \text{ survives}; \text{right}(l) \leftarrow \text{right}(r) + 1$
    - Else $\implies r \text{ survives}; \text{left}(r) \leftarrow \text{left}(l) + 1$

- Analysis ideas:
  - Heavy items fund light items
  - Elimination chain within weight class: $\#\text{items} = \Omega(2^{\#\text{interval}})$
  $\implies O(\log \sigma)$-competitive
Invariants after epoch $q$:

- $|\text{OPT}| = q \cdot |\text{ALG}|$
- $q - 1$ “OPT only” intervals separating “OPT+ALG” intervals
Invariants after epoch $q$:
- $|\text{OPT}| = q \cdot |\text{ALG}|$
- $q - 1$ “OPT only” intervals separating “OPT+ALG” intervals

Epoch 1:
- $n/2$ intervals of length 2
- OPT $= [n] \setminus \text{ALG}$
Invariants after epoch $q$:

- $|\text{OPT}| = q \cdot |\text{ALG}|$
- $q - 1$ “OPT only” intervals separating “OPT+ALG” intervals

Epoch 1:

- $n/2$ intervals of length 2
- $\text{OPT} = [n] \setminus \text{ALG}$

Epoch $q + 1$:
Invariants after epoch $q$:
- $|\text{OPT}| = q \cdot |\text{ALG}|$
- $q - 1$ “OPT only” intervals separating “OPT+ALG” intervals

Epoch 1:
- $n/2$ intervals of length 2
- $\text{OPT} = [n] \setminus \text{ALG}$

Epoch $q + 1$:
Invariants after epoch $q$:
- $|\text{OPT}| = q \cdot |\text{ALG}|$
- $q - 1$ “OPT only” intervals separating “OPT+ALG” intervals

Epoch 1:
- $n/2$ intervals of length 2
- $\text{OPT} = [n] \setminus \text{ALG}$

Epoch $q + 1$:

$\sigma_q \leq 5^q$

#epochs $= \Theta(\log \sigma)$

$|\text{OPT}| = \Omega(\log \sigma \cdot |\text{ALG}|)$
Resource Augmentation

- Online algorithm may select two survivors per conflict
- Compared to offline optimum that selects one survivor
Online algorithm may select two survivors per conflict
Compared to offline optimum that selects one survivor

Algorithm L&R:
In each conflict, select the first and last live tasks
Resource Augmentation

- Online algorithm may select two survivors per conflict
- Compared to offline optimum that selects one survivor

Algorithm L&R:
In each conflict, select the first and last live tasks

Analysis:

Lemma:
Let \( i, i' \in \text{OPT} \) s.t. \( i < i' \). Then, \( \exists j \in \text{L&R} \) s.t. \( j \in [i, i') \).

Proof:
Suppose not. Consider the time when L&R dropped the last task from \( [i, i') \).
Must be due to a conflict strictly containing \( [i, i') \), therefore containing both \( i, i' \), contradiction to \( i, i' \in \text{OPT} \).
Resource Augmentation

- Online algorithm may select two survivors per conflict
- Compared to offline optimum that selects one survivor

Algorithm L&R:
In each conflict, select the first and last live tasks

Analysis:

- **Lemma:**
  Let $i, i' \in \text{OPT}$ s.t. $i < i'$. Then, $\exists j \in L&R$ s.t. $j \in [i, i')$.
  
  **Proof:**
  Suppose not. Consider the time when L&R dropped the last task from $[i, i')$.
  Must be due to a conflict strictly containing $[i, i')$, therefore containing both $i, i'$,
  contradiction to $i, i' \in \text{OPT}$

- **Corollary:** $|L&R| \geq |\text{OPT}|$
Conclusion

Our Results:

□ Introduced scheduling with interval conflicts
  – Sequential, Oblivious
  – Contiguous, Non-contiguous

□ $O(\log \sigma)$-competitive algorithms

□ $\Omega(\log \sigma)$ bound on competitive ratio

□ $\Omega(n)$ bound for oblivious non-contiguous model
Conclusion

Our Results:
- Introduced scheduling with interval conflicts
  - Sequential, Oblivious
  - Contiguous, Non-contiguous
- \( O(\log \sigma) \)-competitive algorithms
- \( \Omega(\log \sigma) \) bound on competitive ratio
- \( \Omega(n) \) bound for oblivious non-contiguous model

Open Questions:
- Randomized algorithms?
- Conflict capacities?
  - Lower bound
  - Non-uniform capacities