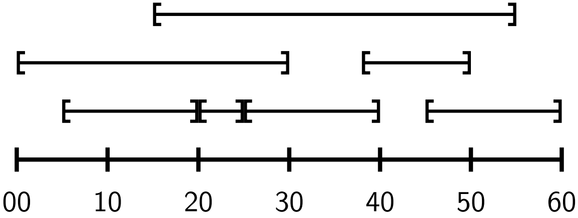
Correctness proofs for interval scheduling
COMS20010 2020, Video lecture 2-2

John Lapinskas, University of Bristol

Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

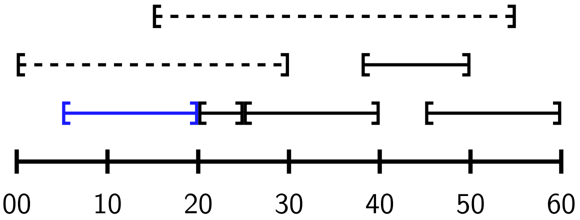
Goal: Prove our pseudocode $\textsc{GreedySchedule}$ is correct.



Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

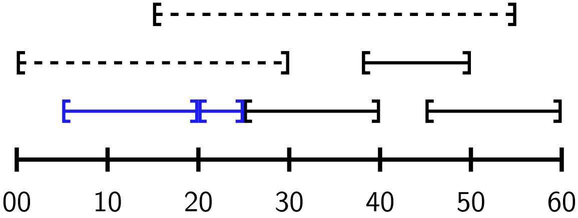
Goal: Prove our pseudocode $\textsc{GreedySchedule}$ is correct.



Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

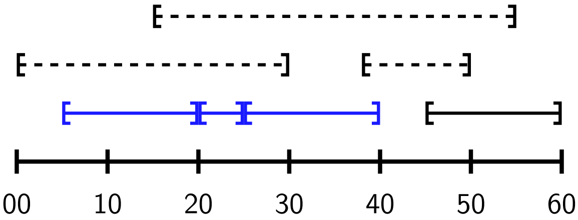
Goal: Prove our pseudocode $\textsc{GreedySchedule}$ is correct.



Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

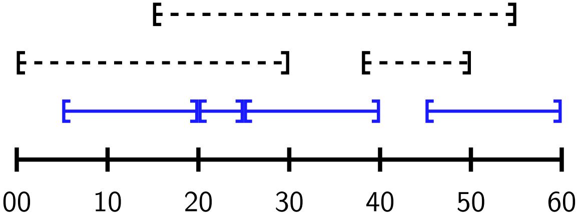
Goal: Prove our pseudocode GreedySchedule is correct.



Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

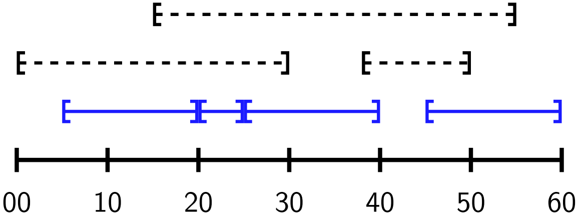
Goal: Prove our pseudocode $\textsc{GreedySchedule}$ is correct.



Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

Goal: Prove our pseudocode $\textsc{GreedySchedule}$ is correct.



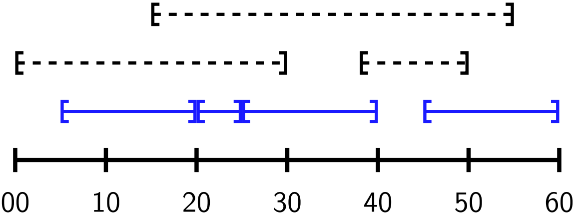
Let's define this formally: breaking ties arbitrarily, let

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

Goal: Prove our pseudocode GreedySchedule is correct.



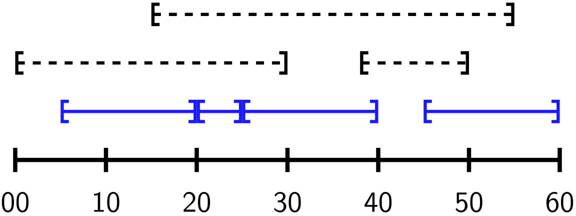
Let's define this formally: breaking ties arbitrarily, let

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\},$$

Recall from last time

To solve interval scheduling with input $\mathcal{R}$, we repeatedly choose the compatible interval with the earliest finish time and add it to the output.

Goal: Prove our pseudocode GREEDYSCHEDULE is correct.



Let's define this formally: breaking ties arbitrarily, let

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

So (we think) GREEDYSCHEDULE calculates $A_0, \dots, A_t$ and outputs A_t. Much easier to work from this than pseudocode when proving correctness!

Step 1: Express GREEDYSCHEDULE mathematically

$$A^+ := \arg\min\{f \colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i \colon A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Step 1: Express GreedySchedule mathematically

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Algorithm: GreedySchedule

Input : An array $\mathcal{R}$ of n requests.
Output : Maximum compatible subset of $\mathcal{R}$.

begin
  Sort $\mathcal{R}$'s entries so that $\mathcal{R} \leftarrow [(s_1, f_1), \dots, (s_n, f_n)]$ where $f_1 \leq \cdots \leq f_n$.
  Initialise $A \leftarrow [\,]$, `lastf` $\leftarrow 0$.
  foreach $i \in \{1, \dots, n\}$ **do**
    if $s_i \geq$ `lastf` **then**
      Append (s_i, f_i) to A and update `lastf` $\leftarrow f_i$.
  Return A.

Step 1: Express GreedySchedule mathematically

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Algorithm: GreedySchedule

Input : An array $\mathcal{R}$ of n requests.
Output : Maximum compatible subset of $\mathcal{R}$.

```
begin
    Sort R's entries so that
      R ← [(s_1, f_1), ..., (s_n, f_n)] where f_1 ≤ ··· ≤ f_n.
    Initialise A ← [], lastf ← 0.
    foreach i ∈ {1, ..., n} do
        if s_i ≥ lastf then
            Append (s_i, f_i) to A and update
              lastf ← f_i.
    Return A.
```

Lemma: GreedySchedule always outputs A_t.

Step 1: Express GreedySchedule mathematically

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Algorithm: GreedySchedule

Input : An array $\mathcal{R}$ of n requests.
Output : Maximum compatible subset of $\mathcal{R}$.

```
1 begin
2     Sort R's entries so that
        R ← [(s_1, f_1), ..., (s_n, f_n)] where f_1 ≤ ··· ≤ f_n.
3     Initialise A ← [], lastf ← 0.
4     foreach i ∈ {1, ..., n} do
5         if s_i ≥ lastf then
6             Append (s_i, f_i) to A and update
                lastf ← f_i.
7     Return A.
```

Lemma: GreedySchedule always outputs A_t.

Proof: By induction from the following loop invariant. At the start of the i'th iteration of 4–7:

- A is equal to $A_t \cap \{(s_1, f_1), \dots, (s_{i-1}, f_{i-1})\}$;
- `lastf` is equal to the latest finish time of any request in A (or 0 if $A = []$).

Step 1: Express GreedySchedule mathematically

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Algorithm: GreedySchedule

Input : An array $\mathcal{R}$ of n requests.
Output : Maximum compatible subset of $\mathcal{R}$.

```
1 begin
2     Sort R's entries so that
        R ← [(s_1, f_1), ..., (s_n, f_n)] where f_1 ≤ ··· ≤ f_n.
3     Initialise A ← [], lastf ← 0.
4     foreach i ∈ {1, ..., n} do
5         if s_i ≥ lastf then
6             Append (s_i, f_i) to A and update
                lastf ← f_i.
7     Return A.
```

Lemma: GreedySchedule always outputs A_t.

Proof: By induction from the following loop invariant. At the start of the i'th iteration of 4–7:

- A is equal to $A_t \cap \{(s_1, f_1), \dots, (s_{i-1}, f_{i-1})\}$;
- `lastf` is equal to the latest finish time of any request in A (or 0 if $A = []$).

Base case:

Step 1: Express GreedySchedule mathematically

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Algorithm: GreedySchedule

Input : An array $\mathcal{R}$ of n requests.
Output : Maximum compatible subset of $\mathcal{R}$.

```
1 begin
2     Sort R's entries so that
        R ← [(s_1, f_1), ..., (s_n, f_n)] where f_1 ≤ ··· ≤ f_n.
3     Initialise A ← [], lastf ← 0.
4     foreach i ∈ {1, ..., n} do
5         if s_i ≥ lastf then
6             Append (s_i, f_i) to A and update
                lastf ← f_i.
7     Return A.
```

Lemma: GreedySchedule always outputs A_t.

Proof: By induction from the following loop invariant. At the start of the i'th iteration of 4–7:

- A is equal to $A_t \cap \{(s_1, f_1), \dots, (s_{i-1}, f_{i-1})\}$;
- `lastf` is equal to the latest finish time of any request in A (or 0 if $A = []$).

Base case: ✓

Step 1: Express GreedySchedule mathematically

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Algorithm: GreedySchedule

Input : An array $\mathcal{R}$ of n requests.
Output : Maximum compatible subset of $\mathcal{R}$.

```
1 begin
2     Sort R's entries so that
        R ← [(s_1, f_1), ..., (s_n, f_n)] where f_1 ≤ ··· ≤ f_n.
3     Initialise A ← [], lastf ← 0.
4     foreach i ∈ {1, ..., n} do
5         if s_i ≥ lastf then
6             Append (s_i, f_i) to A and update
                lastf ← f_i.
7     Return A.
```

Lemma: GreedySchedule always outputs A_t.

Proof: By induction from the following loop invariant. At the start of the i'th iteration of 4–7:

- A is equal to $A_t \cap \{(s_1, f_1), \dots, (s_{i-1}, f_{i-1})\}$;
- `lastf` is equal to the latest finish time of any request in A (or 0 if $A = []$).

Base case: ✓

Inductive step:

Step 1: Express GreedySchedule mathematically

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Intuitive algorithm: Calculate and output A_t.

Algorithm: GreedySchedule

Input : An array $\mathcal{R}$ of n requests.
Output : Maximum compatible subset of $\mathcal{R}$.

```
1 begin
2     Sort R's entries so that
        R ← [(s_1, f_1), ..., (s_n, f_n)] where f_1 ≤ ··· ≤ f_n.
3     Initialise A ← [], lastf ← 0.
4     foreach i ∈ {1, ..., n} do
5         if s_i ≥ lastf then
6             Append (s_i, f_i) to A and update
                lastf ← f_i.
7     Return A.
```

Lemma: GreedySchedule always outputs A_t.

Proof: By induction from the following loop invariant. At the start of the i'th iteration of 4–7:

- A is equal to $A_t \cap \{(s_1, f_1), \dots, (s_{i-1}, f_{i-1})\}$;
- `lastf` is equal to the latest finish time of any request in A (or 0 if $A = []$).

Base case: $\checkmark$

Inductive step: $\square$

Step 2: Prove our algorithm outputs a compatible set

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Step 2: Prove our algorithm outputs a compatible set

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Lemma: A_t is a compatible set.

Proof: Instant by induction; A_0 is compatible, and if A_i is compatible then so is $A_{i+1} = A_i \cup \{A^+\}$ by the definition of A_i^+. □

Step 2: Prove our algorithm outputs a compatible set

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Lemma: A_t is a compatible set.

Proof: Instant by induction; A_0 is compatible, and if A_i is compatible then so is $A_{i+1} = A_i \cup \{A^+\}$ by the definition of A_i^+. □

Sometimes, life is easy!

(Without the lemma, this would have needed a tedious loop invariant...)

Step 3: Prove our algorithm outputs a maximum set

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓
Lemma: A_t is a compatible set. ✓

Lemma: A_t is a maximum compatible subset of $\mathcal{R}$.

Step 3: Prove our algorithm outputs a maximum set

$$A^+ := \arg\min\{f\colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i\colon A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓
Lemma: A_t is a compatible set. ✓

Lemma: A_t is a maximum compatible subset of $\mathcal{R}$.

Proof: This is harder! But there's a trick: prove that if we compare A_t to any other compatible set, A_t will always "do better" on a request-by-request basis (not just overall). We can prove this by induction.

Step 3: Prove our algorithm outputs a maximum set

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓
Lemma: A_t is a compatible set. ✓

Lemma: A_t is a maximum compatible subset of $\mathcal{R}$.

Proof: This is harder! But there's a trick: prove that if we compare A_t to any other compatible set, A_t will always "do better" on a request-by-request basis (not just overall). We can prove this by induction.

More formally, let $B \subseteq \mathcal{R}$ be any other compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B.

Then we will show by induction that for all $0 \leq i \leq t$, the last finish time of B_i is no earlier than the last finish time of A_i.

$$A^+ := \arg\min\{f\colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

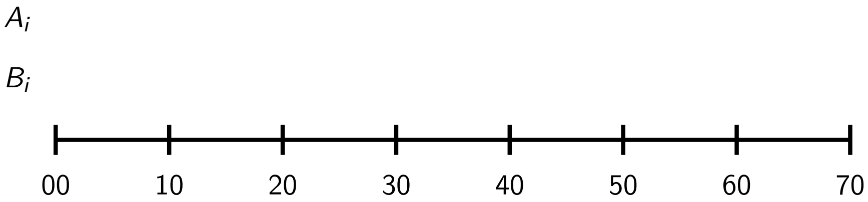
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i\colon A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$
$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i.

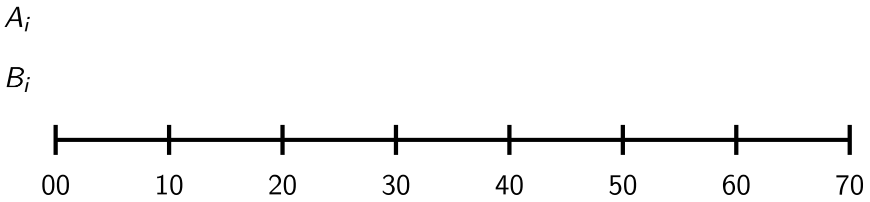


$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i.



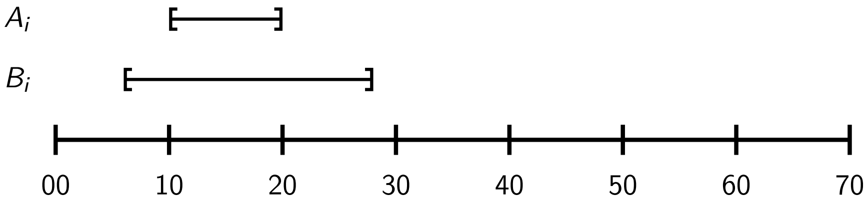
Base case: A_0^+ is the fastest-finishing request in $\mathcal{R}$ by definition. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i.



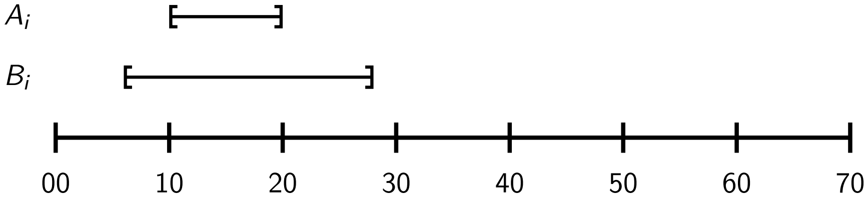
Base case: A_0^+ is the fastest-finishing request in $\mathcal{R}$ by definition. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i. **Base case $i = 1$:** ✓

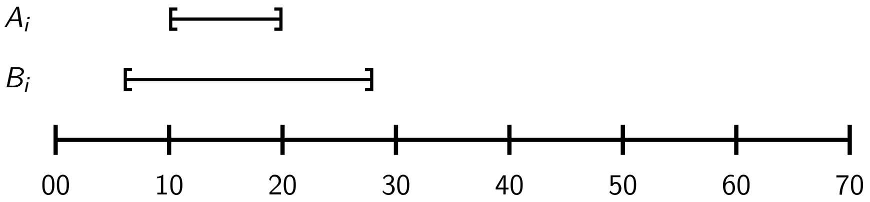


$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i. **Base case $i = 1$:** ✓



Inductive step: Suppose A_i finishes faster than B_i.

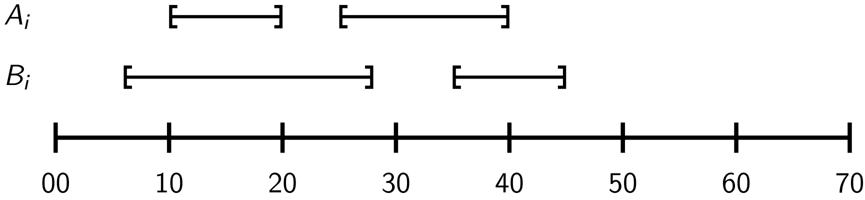
Let B_i^+ be the $(i+1)'$st fastest-finishing element of B.
Since A_i finishes faster than B_i, $A_i \cup \{B_i^+\}$ is compatible.
Hence by definition, A_i^+ exists and finishes no later than B_i^+. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i. **Base case $i = 1$:** ✓



Inductive step: Suppose A_i finishes faster than B_i.

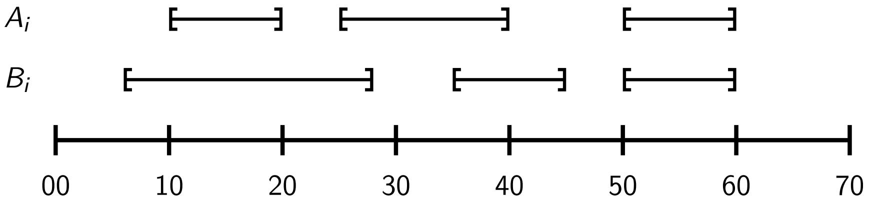
Let B_i^+ be the $(i+1)'$st fastest-finishing element of B.
Since A_i finishes faster than B_i, $A_i \cup \{B_i^+\}$ is compatible.
Hence by definition, A_i^+ exists and finishes no later than B_i^+. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i. **Base case $i = 1$:** ✓



Inductive step: Suppose A_i finishes faster than B_i.

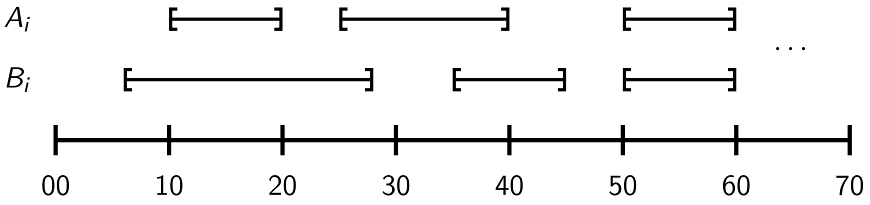
Let B_i^+ be the $(i+1)'$st fastest-finishing element of B.
Since A_i finishes faster than B_i, $A_i \cup \{B_i^+\}$ is compatible.
Hence by definition, A_i^+ exists and finishes no later than B_i^+. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i.

Proof: By induction on i. **Base case $i = 1$:** ✓



Inductive step: Suppose A_i finishes faster than B_i.

Let B_i^+ be the $(i+1)'$st fastest-finishing element of B.
Since A_i finishes faster than B_i, $A_i \cup \{B_i^+\}$ is compatible.
Hence by definition, A_i^+ exists and finishes no later than B_i^+. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Lemma: A_t is a compatible set. ✓

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Lemma: A_t is a compatible set. ✓

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i. ✓

Putting it all together, we obtain...

Theorem: GreedySchedule outputs A_t, which is a **maximum** compatible set. □

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Lemma: A_t is a compatible set. ✓

Lemma: Let $B \subseteq \mathcal{R}$ be any compatible set with $|B| \geq |A_t|$, and let B_i consist of the i fastest-finishing elements of B. Then $i \leq t$, and the last finish time of B_i is no earlier than the last finish time of A_i. ✓

Putting it all together, we obtain...

Theorem: GreedySchedule outputs A_t, which is a **maximum** compatible set. □

This technique of proving that the greedy solution "stays ahead" of any other solution is very useful for other greedy algorithms as well!

An alternative proof of optimality

$$A^+ := \arg\min\{f \colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i \colon A_i \text{ is defined}\}.$$

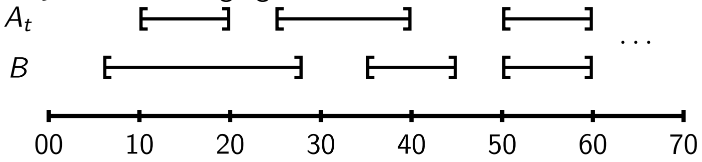
An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

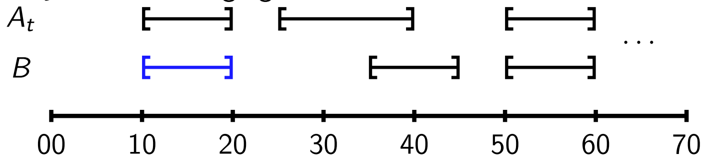


An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

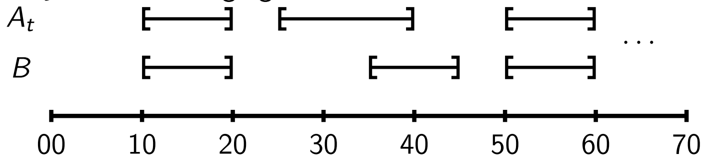


An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

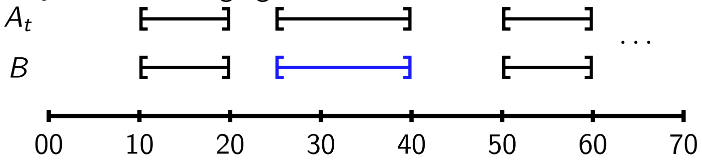


An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

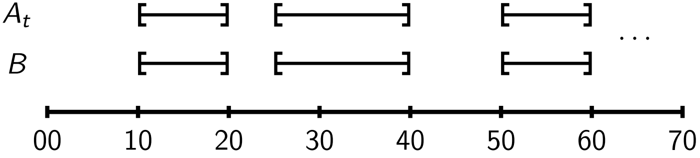


An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

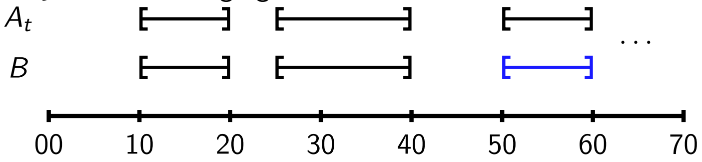


An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

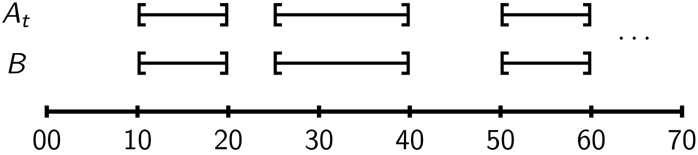


An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

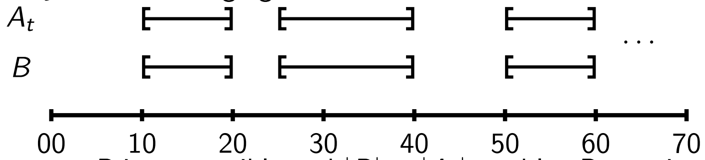


An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

An alternative method: show that any maximum compatible set B can be **turned into** A_t without changing the number of intervals.

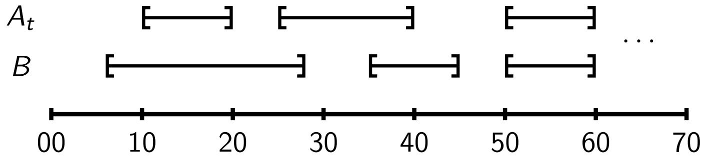


Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible.

An alternative proof of optimality

$$A^+ := \arg\min\{f\colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i\colon A_i \text{ is defined}\}.$$



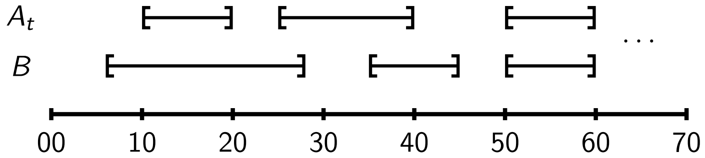
Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible.

Proof: By induction on i.

An alternative proof of optimality

$$A^+ := \arg\min\{f\colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i\colon A_i \text{ is defined}\}.$$



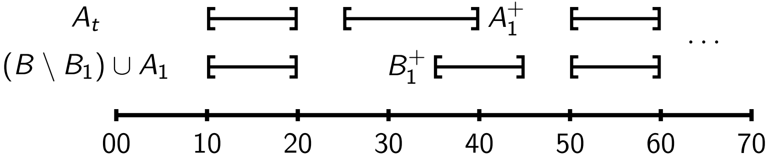
Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible.

Proof: By induction on i. **Base case:** Immediate for $i = 0$. ✓

An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$



Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible.

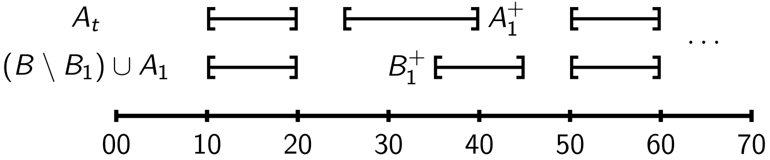
Proof: By induction on i. **Base case:** Immediate for $i = 0$. ✓

Inductive step: Suppose $(B \setminus B_i) \cup A_i$ is compatible, write $B_{i+1} \setminus B_i = \{B_i^+\}$. Then we are done if A_i^+ is compatible with A_i and $B \setminus B_{i+1}$.

An alternative proof of optimality

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$



Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible.

Proof: By induction on i. **Base case:** Immediate for $i = 0$. ✓

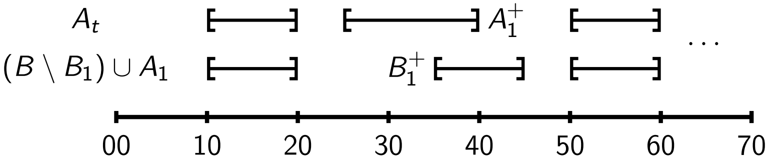
Inductive step: Suppose $(B \setminus B_i) \cup A_i$ is compatible, write $B_{i+1} \setminus B_i = \{B_i^+\}$. Then we are done if A_i^+ is compatible with A_i and $B \setminus B_{i+1}$.

A_i^+ is compatible with A_i by definition.

An alternative proof of optimality

$$A^+ := \arg\min\{f \colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i \colon A_i \text{ is defined}\}.$$



Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible.

Proof: By induction on i. **Base case:** Immediate for $i = 0$. ✓

Inductive step: Suppose $(B \setminus B_i) \cup A_i$ is compatible, write $B_{i+1} \setminus B_i = \{B_i^+\}$. Then we are done if A_i^+ is compatible with A_i and $B \setminus B_{i+1}$.

A_i^+ is compatible with A_i by definition. By induction, $B \setminus B_i \cup A_i$ is compatible, so B_i^+ is compatible with A_i, so A_i^+ finishes earlier than B_i^+ by definition. Hence A_i^+ is also compatible with $B \setminus B_{i+1}$. ✓

$$A^+ := \arg\min\{f : (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i : A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Lemma: A_t is a compatible set. ✓

Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible. ✓

Theorem: A_t is a **maximum** compatible set.

$$A^+ := \arg\min\{f\colon (s,f) \in \mathcal{R}, A \cup \{(s,f)\} \text{ is compatible}\} \text{ for all } A \subseteq \mathcal{R},$$

$$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}, \qquad t := \max\{i\colon A_i \text{ is defined}\}.$$

Lemma: GreedySchedule outputs A_t. ✓

Lemma: A_t is a compatible set. ✓

Lemma: Suppose B is compatible and $|B| \geq |A_t|$, and let B_i consist of the $i \geq 0$ fastest-finishing elements of B. Then $(B \setminus B_i) \cup A_i$ is compatible. ✓

Theorem: A_t is a **maximum** compatible set.

On taking $i = t$, we see that $(B \setminus B_t) \cup A_t$ is compatible — i.e. we can remove the first t intervals from B and replace them with the whole of A_t.

Since A_t is **maximal** — that is, since we can't add any intervals to A_t and keep it compatible — it follows that $|B| = |A_t|$.

(**Exercise:** Prove by induction that A_t is maximal...)

Choosing between the two methods

Both types of argument used this lecture, "**greedy stays ahead** proofs" and "**exchange** proofs", are powerful and widely-used.

Sometimes only one approach will work easily, but often (like here) the two approaches feel like they are doing the same thing under the surface. Use whichever one you find more natural — it's a matter of taste!

Choosing between the two methods

Both types of argument used this lecture, "**greedy stays ahead** proofs" and "**exchange** proofs", are powerful and widely-used.

Sometimes only one approach will work easily, but often (like here) the two approaches feel like they are doing the same thing under the surface. Use whichever one you find more natural — it's a matter of taste!