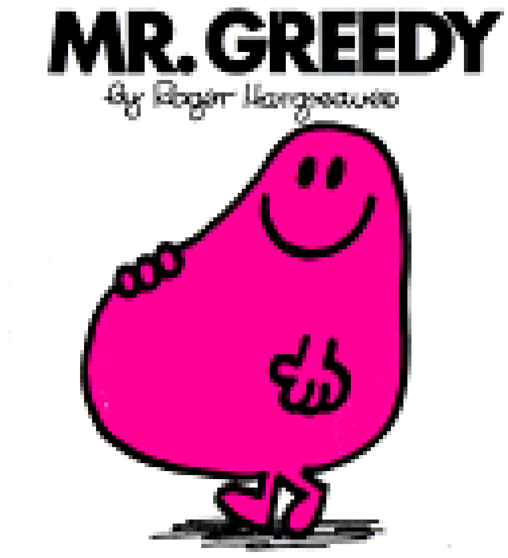
4.1 Interval Scheduling

4.1 Interval Scheduling

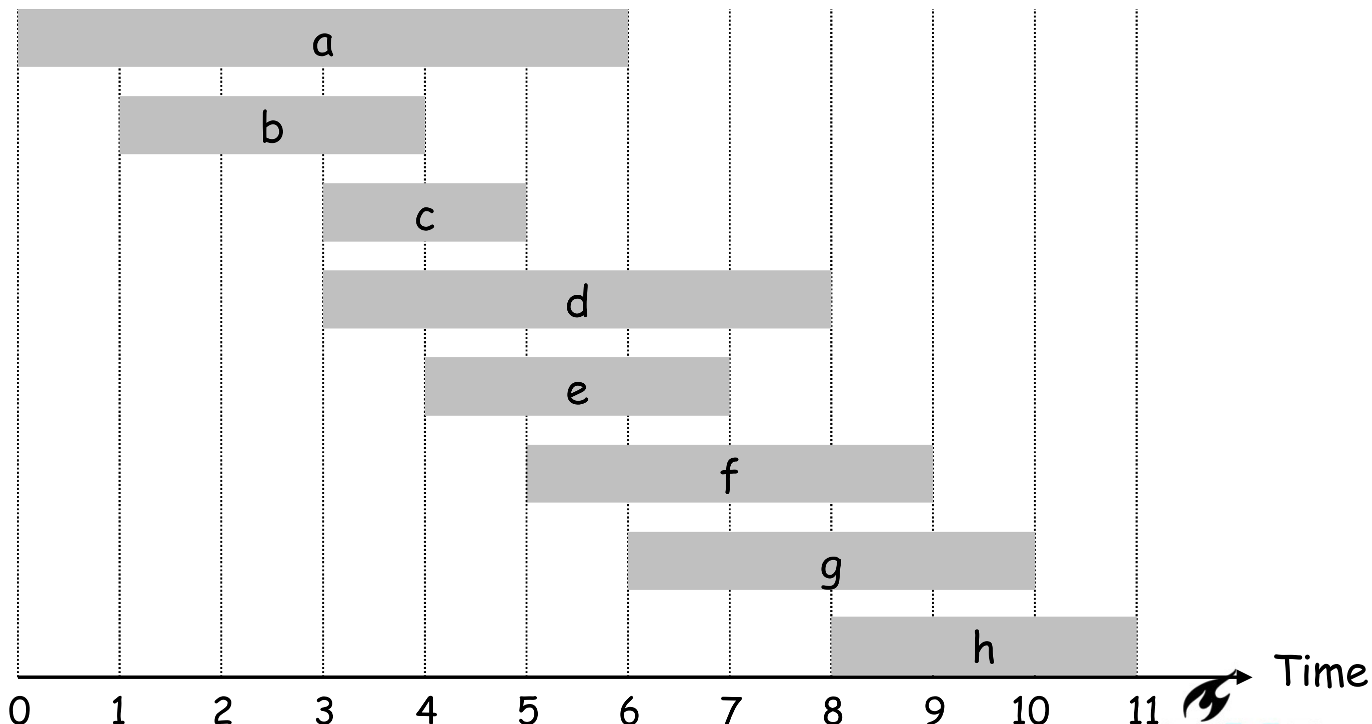


Ref: Mr. Greedy is part of the Mr. Men series of books, by Roger Hargreaves.

Interval scheduling (activity selection)

- Job j starts at s_j and finishes at f_j.
- Two jobs compatible if they don't overlap.

Q. What is the maximum subset of mutually compatible jobs?

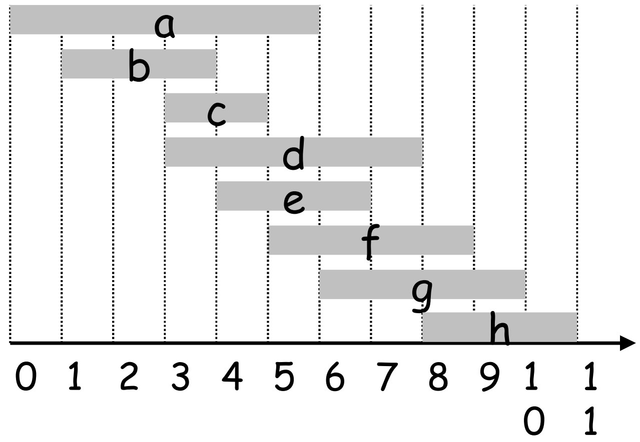


Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some order. Take each job provided it's compatible with the ones already taken.

- [Earliest start time] Consider jobs in ascending order of start time s_j.
- [Earliest finish time] Consider jobs in ascending order of finish time f_j.
- [Shortest interval] Consider jobs in ascending order of interval length $f_j - s_j$.
- [Fewest conflicts] For each job, count the number of conflicting jobs c_j. Schedule in ascending order of conflicts c_j.

Q. Which one do you think may work? (2 min)



Interval Scheduling: Greedy Algorithms

Greedy template. Consider jobs in some order. Take each job provided it's compatible with the ones already taken.

earliest start time?

shortest interval?

fewest conflicts?

Interval Scheduling: Greedy Algorithm

Greedy algorithm. Consider jobs in increasing order of finish time. Take each job provided it's compatible with the ones already taken.

```
Sort jobs by finish times so that f₁ ≤ f₂ ≤ ... ≤ fₙ.

  jobs selected

A ← φ
for j = 1 to n {
   if (job j compatible with A)
      A ← A ∪ {j}
}
return A
```

Implementation. O(n log n).

- Remember job j* that was added last to A.
- Job j is compatible with A if $s_j \geq f_{j^*}$.

TU Delft

Invariant (proof by induction)

Lemma. Greedy algorithm is sound (i.e., all jobs in A are compatible).

Pf. (by induction: using an invariant)

Q. What are the basic elements of a proof by induction?

TU Delft

Invariant (proof by induction)

Lemma. Greedy algorithm is sound (i.e., all jobs in A are compatible).

Pf. (by induction: using an invariant)

Q. What are the basic elements of a proof by induction?

Base: (Initialization) When A=ϕ then all jobs in A are trivially compatible.

(Maintenance)

Hypothesis (IH): All jobs $i < j$ in A are compatible.

Step: To prove: all jobs $i < j+1$ in A are compatible.

TU Delft

Invariant (proof by induction)

Lemma. Greedy algorithm is sound (i.e., all jobs in A are compatible).

Pf. (by induction: using an invariant)

Q. What are the basic elements of a proof by induction?

Base: (Initialization) When A=ϕ then all jobs in A are trivially compatible.

(Maintenance)

Hypothesis (IH): All jobs i < j in A are compatible.

Step: To prove: all jobs i < j+1 in A are compatible.

Given is that all jobs i<j in A are compatible.

If j is not in A then it follows that all jobs i < j+1 in A are compatible.

Otherwise, j was inserted in A and thus condition "**`job j compatible with A`**" holds.

Thus in both cases all jobs i < j+1 in A are compatible.

Invariant (proof by induction)

Lemma. Greedy algorithm is sound (i.e., all jobs in A are compatible).

Pf. (by induction: using an invariant)

Q. What are the basic elements of a proof by induction?

Base: (Initialization) When A=ϕ then all jobs in A are trivially compatible.

(Maintenance)

Hypothesis (IH): All jobs i < j in A are compatible.

Step: To prove: all jobs i < j+1 in A are compatible.

Given is that all jobs i<j in A are compatible.

If j is not in A then it follows that all jobs i < j+1 in A are compatible.

Otherwise, j was inserted in A and thus condition "**`job j compatible with A`**" holds.

Thus in both cases all jobs i < j+1 in A are compatible.

(Termination; sometimes you can use the negation of a while here as well)

Conclusion: With induction (till j=n), all jobs (i<n+1) in A are compatible.▪

Interval Scheduling: Analysis

Theorem 4.3. Greedy algorithm is optimal.

Pf. (by contradiction: exchange argument)

Q. How do we start a proof by contradiction?

From the "Proving guide" (Blackboard)

In order to prove a proposition P by contradiction:

1. Write, "We use proof by contradiction."
2. Write, "Suppose P is false."
3. Deduce a logical contradiction.
4. Write, "This is a contradiction. Therefore, P must be true."

The equivalent structure in a Fitch proof is as follows:

1	$\quad\mid\ \neg P$	(hypothesis)
2	$\quad\mid\ \vdots$	
3	$\quad\mid\ Q$	
4	$\quad\mid\ \neg Q$	
5	$\neg\neg P$	($\neg$-intro, 1,3,4)
6	P	($\neg$-elim, 5)

P = Greedy is optimal.

Interval Scheduling: Analysis

Theorem 4.3. Greedy algorithm is optimal.

Pf. (by contradiction: exchange argument)

Suppose Greedy is not optimal.

Interval Scheduling: Analysis

Theorem 4.3. Greedy algorithm is optimal.

Pf. (by contradiction: exchange argument)

Suppose Greedy is not optimal.

Q. How can we arrive at a contradiction?

A. See where the optimal solution is different from Greedy.

TU Delft

Interval Scheduling: Analysis

Theorem 4.3. Greedy algorithm is optimal.

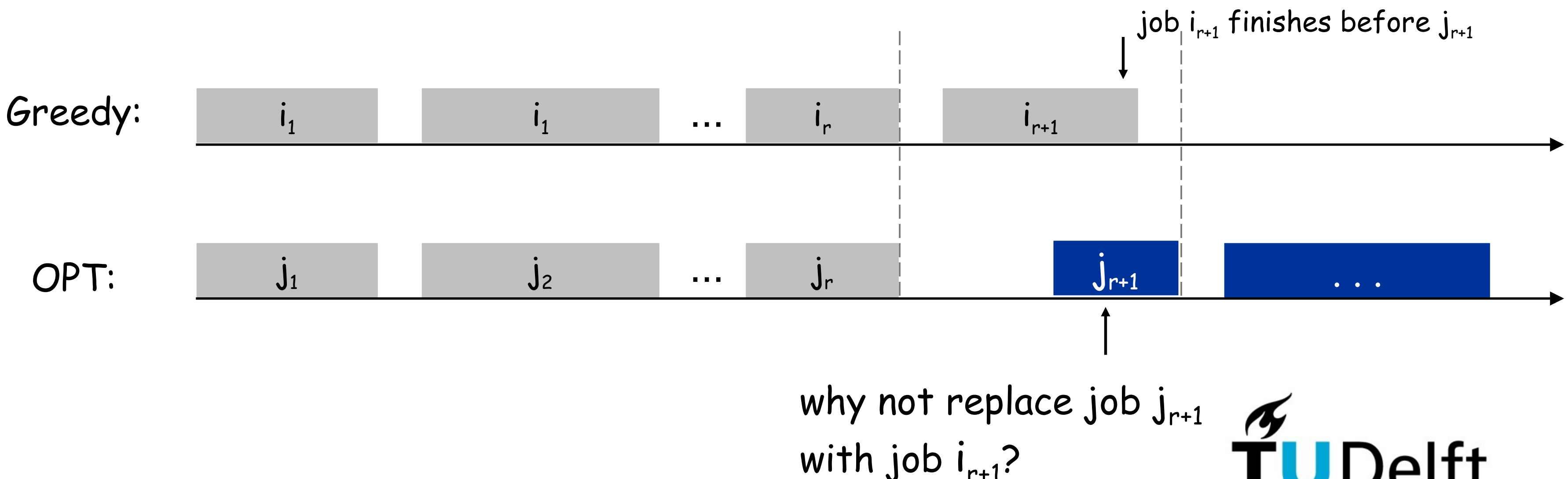
Pf. (by contradiction: exchange argument)

Suppose Greedy is not optimal.

Let $i_1, i_2, \ldots i_k$ denote set of jobs selected by Greedy.

Let $j_1, j_2, \ldots j_m$ denote set of jobs in the optimal solution.

Consider OPT solution that follows Greedy as long as possible (up to r), so with $i_1 = j_1$, $i_2 = j_2$, ..., $i_r = j_r$ for the largest possible value of r.



Interval Scheduling: Analysis

Theorem 4.3. Greedy algorithm is optimal.

Pf. (by contradiction: exchange argument)

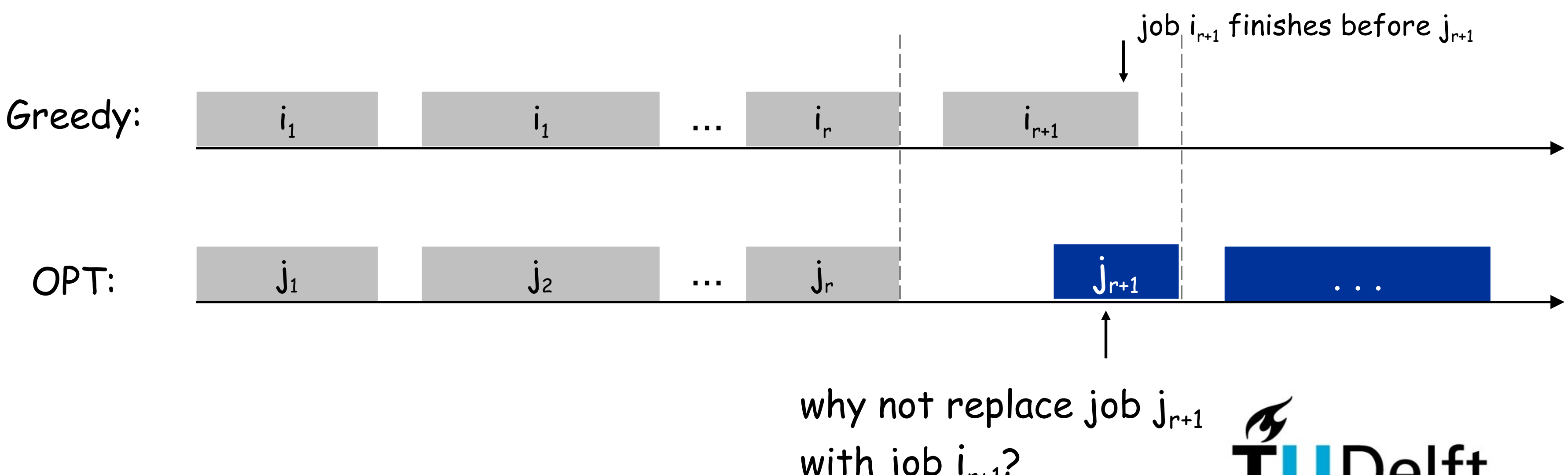
Suppose Greedy is not optimal.

Let $i_1, i_2, \ldots i_k$ denote set of jobs selected by Greedy.

Let $j_1, j_2, \ldots j_m$ denote set of jobs in the optimal solution.

Consider OPT solution that follows Greedy as long as possible (up to r), so with $i_1 = j_1$, $i_2 = j_2$, ..., $i_r = j_r$ for the largest possible value of r.

Q. Where is the contradiction?



Interval Scheduling: Analysis

Proof in book (p120-121) is a bit more formal, relying on a proof by induction.

Theorem 4.3. Greedy algorithm is optimal.

Pf. (by contradiction: exchange argument)

Suppose Greedy is not optimal.

Let $i_1, i_2, \ldots i_k$ denote set of jobs selected by Greedy.

Let $j_1, j_2, \ldots j_m$ denote set of jobs in the optimal solution.

Consider OPT solution that follows Greedy as long as possible (up to r), so with $i_1 = j_1, i_2 = j_2, \ldots, i_r = j_r$ for the largest possible value of r.

Consider then first choice that is different.

Change OPT to OPT': still optimal, but follows Greedy longer.

Contradiction: OPT' follows Greedy longer than OPT! ▪

job i_{r+1} finishes before j_{r+1}

solution still feasible and optimal, but contradicts maximality of r.

Proof by induction ("Greedy stays ahead")

Lemma 4.2. For all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$. (i for Greedy; j for OPT)

Pf. (by induction: Greedy stays ahead)

Q. What are the basic elements of a proof by induction?

Proof by induction ("Greedy stays ahead")

Lemma 4.2. For all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$. (i for Greedy; j for OPT)

Pf. (by induction: Greedy stays ahead)

Base: When k=1,

Hypothesis (IH): Suppose that for all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$.

Step: To prove: for all $r \leq k+1$ it holds that $f(i_r) \leq f(j_r)$.

Proof by induction ("Greedy stays ahead")

Lemma 4.2. For all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$. (i for Greedy; j for OPT)

Pf. (by induction: Greedy stays ahead)
Base: When k=1, r=1, so the only job i_1 is chosen such that $f(i_1) \leq f(j_1)$.

Hypothesis (IH): Suppose that for all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$.
Step: To prove: for all $r \leq k+1$ it holds that $f(i_r) \leq f(j_r)$.

TU Delft

Proof by induction ("Greedy stays ahead")

Lemma 4.2. For all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$. (i for Greedy; j for OPT)

Pf. (by induction: Greedy stays ahead)

Base: When $k=1$, $r=1$, so the only job i_1 is chosen such that $f(i_1) \leq f(j_1)$.

Hypothesis (IH): Suppose that for all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$.

Step: To prove: for all $r \leq k+1$ it holds that $f(i_r) \leq f(j_r)$.

For all $r \leq k$ this follows immediately from the IH.

Consider $r=k+1$.

Q. How can we conclude that $f(i_{k+1}) \leq f(j_{k+1})$?

Proof by induction ("Greedy stays ahead")

Lemma 4.2. For all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$. (i for Greedy; j for OPT)

Pf. (by induction: Greedy stays ahead)

Base: When $k=1$, $r=1$, so the only job i_1 is chosen such that $f(i_1) \leq f(j_1)$.

Hypothesis (IH): Suppose that for all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$.

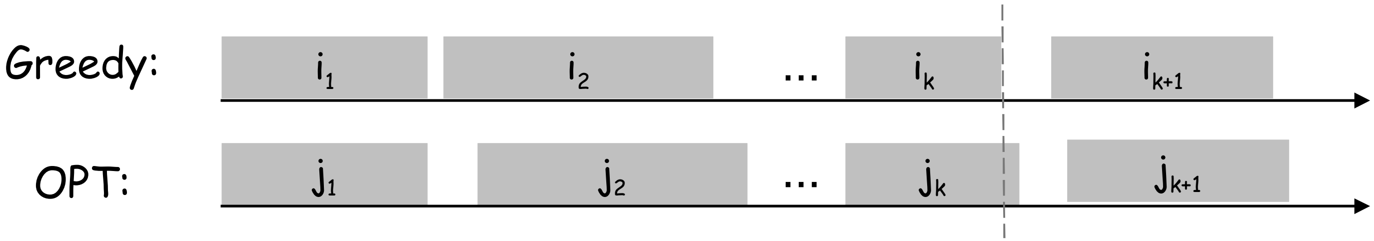
Step: To prove: for all $r \leq k+1$ it holds that $f(i_r) \leq f(j_r)$.

For all $r \leq k$ this follows immediately from the IH.

Consider $r=k+1$.

Q. How can we conclude that $f(i_{k+1}) \leq f(j_{k+1})$?

Q. From which jobs can Greedy choose?



Proof by induction ("Greedy stays ahead")

Lemma 4.2. For all r≤k it holds that $f(i_r) \leq f(j_r)$. (i for Greedy; j for OPT)

Pf. (by induction: Greedy stays ahead)

Base: When k=1, r=1, so the only job i_1 is chosen such that $f(i_1) \leq f(j_1)$.

Hypothesis (IH): Suppose that for all r≤k it holds that $f(i_r) \leq f(j_r)$.

Step: To prove: for all r≤k+1 it holds that $f(i_r) \leq f(j_r)$.

For all r≤k this follows immediately from the IH.

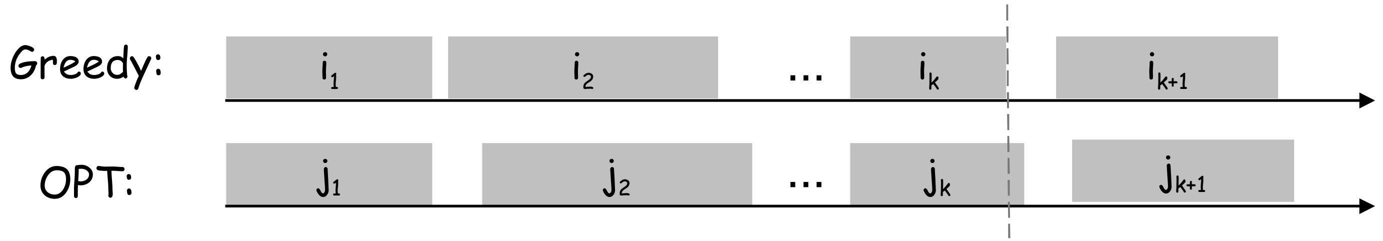
Consider r=k+1.

We know that $f(j_k) \leq s(j_{k+1})$ (in OPT).

So $f(i_k) \leq s(j_{k+1})$ with the IH.

So j_{k+1} can also be chosen by Greedy (i_{k+1} can be equal to j_{k+1}).

Greedy chooses job with smallest end time. Therefore $f(i_{k+1}) \leq f(j_{k+1})$.



This proof can be found on page 120. Requires the (brief) proof of 4.3 on page 121 to show that Greedy is optimal.

TU Delft

Interval Scheduling: Analysis

Theorem 4.3. Greedy algorithm is optimal.

Pf. (by contradiction)

Let i_1, i_2, ... i_k denote set of jobs selected by Greedy.

Let j_1, j_2, ... j_m denote set of jobs in the optimal solution.

Suppose Greedy is not optimal, thus $k < m$.

However, for all $r \leq k$ it holds that $f(i_r) \leq f(j_r)$ by Lemma 4.2.

In particular, $f(i_k) \leq f(j_k)$.

But then there is a job j_{k+1}, which starts after j_k and thus i_k ends.

But then after Greedy inserted i_k, there was another compatible job left.

Contradiction with Greedy schedule having only k jobs. ▪

This proof can be found on page 121.