

7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

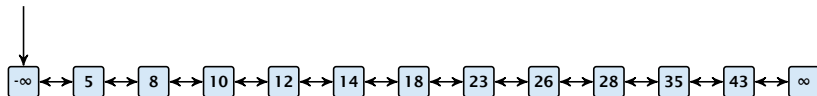
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

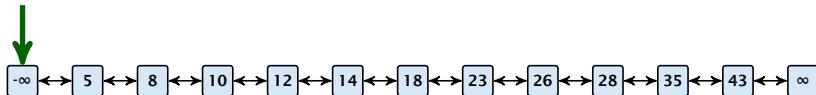
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

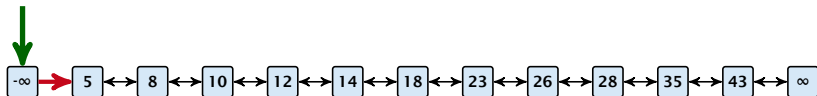
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

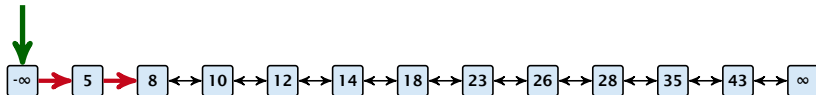
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

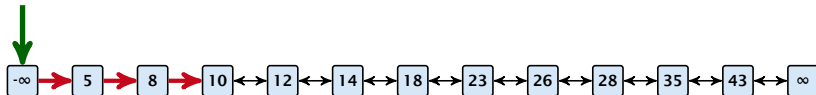
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

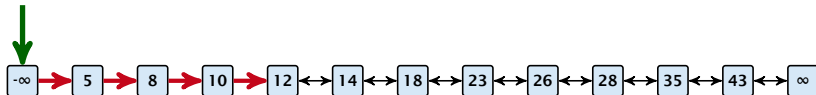
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

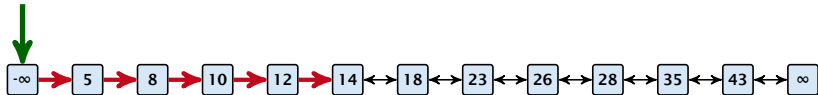
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

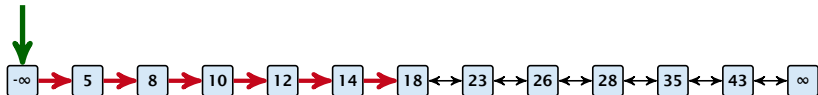
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

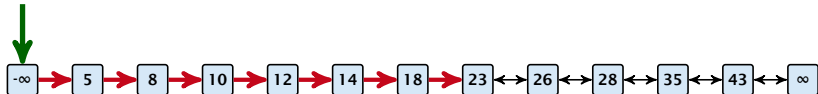
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

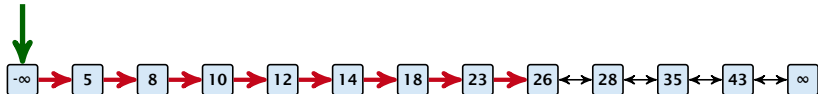
- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

Why do we not use a list for implementing the ADT Dynamic Set?

- ▶ time for search $\Theta(n)$
- ▶ time for insert $\Theta(n)$ (dominated by searching the item)
- ▶ time for delete $\Theta(1)$ if we are given a handle to the object, otw. $\Theta(n)$



7.6 Skip Lists

How can we improve the search-operation?

7.6 Skip Lists

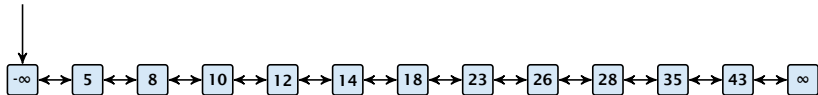
How can we improve the search-operation?

Add an express lane:

7.6 Skip Lists

How can we improve the search-operation?

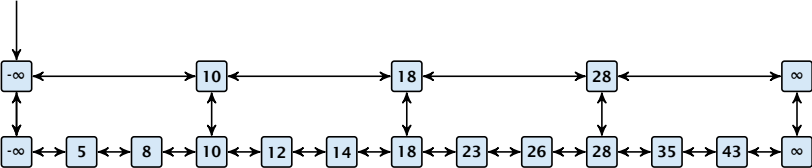
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

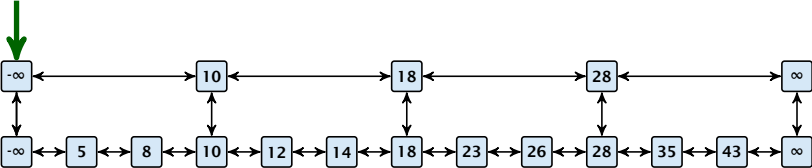
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

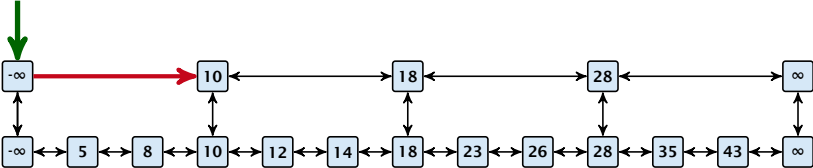
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

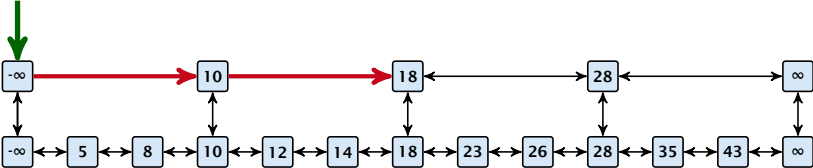
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

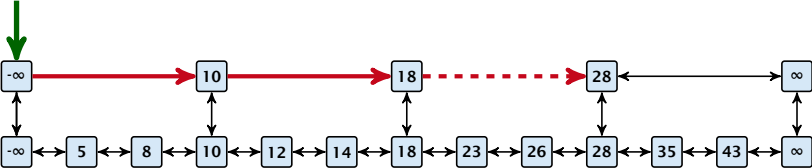
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

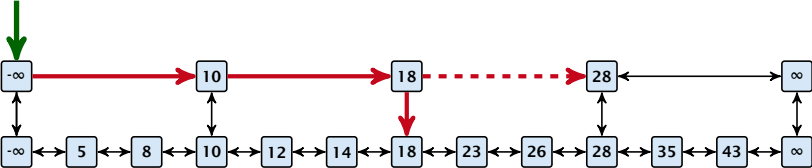
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

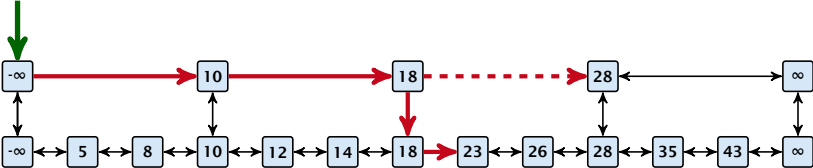
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

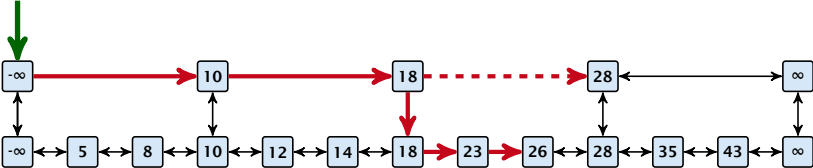
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

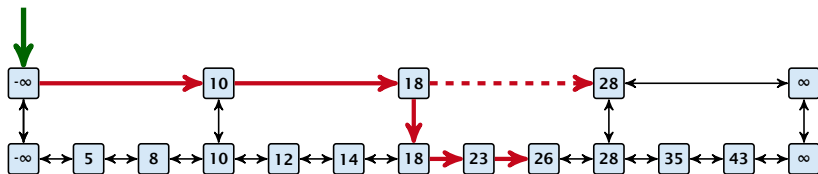
Add an express lane:



7.6 Skip Lists

How can we improve the search-operation?

Add an express lane:

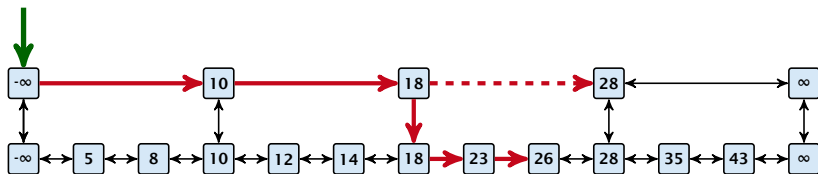


Let $|L_1|$ denote the number of elements in the “express lane”, and $|L_0| = n$ the number of all elements (ignoring dummy elements).

7.6 Skip Lists

How can we improve the search-operation?

Add an express lane:



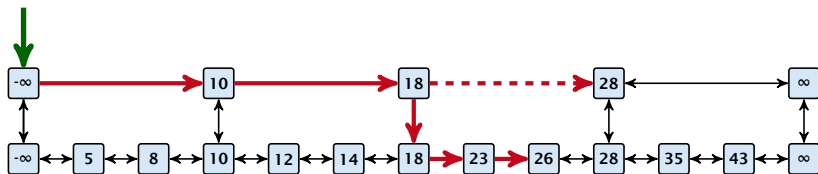
Let $|L_1|$ denote the number of elements in the “express lane”, and $|L_0| = n$ the number of all elements (ignoring dummy elements).

Worst case search time: $|L_1| + \frac{|L_0|}{|L_1|}$ (ignoring additive constants)

7.6 Skip Lists

How can we improve the search-operation?

Add an express lane:



Let $|L_1|$ denote the number of elements in the “express lane”, and $|L_0| = n$ the number of all elements (ignoring dummy elements).

Worst case search time: $|L_1| + \frac{|L_0|}{|L_1|}$ (ignoring additive constants)

Choose $|L_1| = \sqrt{n}$. Then search time $\Theta(\sqrt{n})$.

7.6 Skip Lists

Add more express lanes. Lane L_i contains roughly every $\frac{L_{i-1}}{L_i}$ -th item from list L_{i-1} .

7.6 Skip Lists

Add more express lanes. Lane L_i contains roughly every $\frac{L_{i-1}}{L_i}$ -th item from list L_{i-1} .

Search(x) ($k + 1$ lists L_0, \dots, L_k)

7.6 Skip Lists

Add more express lanes. Lane L_i contains roughly every $\frac{L_{i-1}}{L_i}$ -th item from list L_{i-1} .

Search(x) ($k + 1$ lists L_0, \dots, L_k)

- ▶ Find the largest item in list L_k that is smaller than x . At most $|L_k| + 2$ steps.

7.6 Skip Lists

Add more express lanes. Lane L_i contains roughly every $\frac{L_{i-1}}{L_i}$ -th item from list L_{i-1} .

Search(x) ($k + 1$ lists L_0, \dots, L_k)

- ▶ Find the largest item in list L_k that is smaller than x . At most $|L_k| + 2$ steps.
- ▶ Find the largest item in list L_{k-1} that is smaller than x . At most $\lceil \frac{|L_{k-1}|}{|L_k|+1} \rceil + 2$ steps.

7.6 Skip Lists

Add more express lanes. Lane L_i contains roughly every $\frac{L_{i-1}}{L_i}$ -th item from list L_{i-1} .

Search(x) ($k + 1$ lists L_0, \dots, L_k)

- ▶ Find the largest item in list L_k that is smaller than x . At most $|L_k| + 2$ steps.
- ▶ Find the largest item in list L_{k-1} that is smaller than x . At most $\lceil \frac{|L_{k-1}|}{|L_k|+1} \rceil + 2$ steps.
- ▶ Find the largest item in list L_{k-2} that is smaller than x . At most $\lceil \frac{|L_{k-2}|}{|L_{k-1}|+1} \rceil + 2$ steps.

7.6 Skip Lists

Add more express lanes. Lane L_i contains roughly every $\frac{|L_{i-1}|}{2}$ -th item from list L_{i-1} .

Search(x) ($k + 1$ lists L_0, \dots, L_k)

- ▶ Find the largest item in list L_k that is smaller than x . At most $|L_k| + 2$ steps.
- ▶ Find the largest item in list L_{k-1} that is smaller than x . At most $\lceil \frac{|L_{k-1}|}{2} \rceil + 2$ steps.
- ▶ Find the largest item in list L_{k-2} that is smaller than x . At most $\lceil \frac{|L_{k-2}|}{4} \rceil + 2$ steps.
- ▶ ...

7.6 Skip Lists

Add more express lanes. Lane L_i contains roughly every $\frac{L_{i-1}}{L_i}$ -th item from list L_{i-1} .

Search(x) ($k + 1$ lists L_0, \dots, L_k)

- ▶ Find the largest item in list L_k that is smaller than x . At most $|L_k| + 2$ steps.
- ▶ Find the largest item in list L_{k-1} that is smaller than x . At most $\lceil \frac{|L_{k-1}|}{|L_k|+1} \rceil + 2$ steps.
- ▶ Find the largest item in list L_{k-2} that is smaller than x . At most $\lceil \frac{|L_{k-2}|}{|L_{k-1}|+1} \rceil + 2$ steps.
- ▶ ...
- ▶ At most $|L_k| + \sum_{i=1}^k \frac{L_{i-1}}{L_i} + 3(k + 1)$ steps.

7.6 Skip Lists

Choose ratios between list-lengths evenly, i.e., $\frac{|L_{i-1}|}{|L_i|} = r$, and, hence, $L_k \approx r^{-k}n$.

7.6 Skip Lists

Choose ratios between list-lengths evenly, i.e., $\frac{|L_{i-1}|}{|L_i|} = r$, and, hence, $L_k \approx r^{-k}n$.

Worst case running time is: $\mathcal{O}(r^{-k}n + kr)$.

7.6 Skip Lists

Choose ratios between list-lengths evenly, i.e., $\frac{|L_{i-1}|}{|L_i|} = r$, and, hence, $L_k \approx r^{-k}n$.

Worst case running time is: $\mathcal{O}(r^{-k}n + kr)$.

Choose $r = n^{\frac{1}{k+1}}$. Then

$$r^{-k}n + kr$$

7.6 Skip Lists

Choose ratios between list-lengths evenly, i.e., $\frac{|L_{i-1}|}{|L_i|} = r$, and, hence, $L_k \approx r^{-k}n$.

Worst case running time is: $\mathcal{O}(r^{-k}n + kr)$.

Choose $r = n^{\frac{1}{k+1}}$. Then

$$r^{-k}n + kr = \left(n^{\frac{1}{k+1}}\right)^{-k}n + kn^{\frac{1}{k+1}}$$

7.6 Skip Lists

Choose ratios between list-lengths evenly, i.e., $\frac{|L_{i-1}|}{|L_i|} = r$, and, hence, $L_k \approx r^{-k}n$.

Worst case running time is: $\mathcal{O}(r^{-k}n + kr)$.

Choose $r = n^{\frac{1}{k+1}}$. Then

$$\begin{aligned}r^{-k}n + kr &= \left(n^{\frac{1}{k+1}}\right)^{-k}n + kn^{\frac{1}{k+1}} \\ &= n^{1-\frac{k}{k+1}} + kn^{\frac{1}{k+1}}\end{aligned}$$

7.6 Skip Lists

Choose ratios between list-lengths evenly, i.e., $\frac{|L_{i-1}|}{|L_i|} = r$, and, hence, $L_k \approx r^{-k}n$.

Worst case running time is: $\mathcal{O}(r^{-k}n + kr)$.

Choose $r = n^{\frac{1}{k+1}}$. Then

$$\begin{aligned}r^{-k}n + kr &= \left(n^{\frac{1}{k+1}}\right)^{-k}n + kn^{\frac{1}{k+1}} \\ &= n^{1-\frac{k}{k+1}} + kn^{\frac{1}{k+1}} \\ &= (k+1)n^{\frac{1}{k+1}}.\end{aligned}$$

7.6 Skip Lists

Choose ratios between list-lengths evenly, i.e., $\frac{|L_{i-1}|}{|L_i|} = r$, and, hence, $L_k \approx r^{-k}n$.

Worst case running time is: $\mathcal{O}(r^{-k}n + kr)$.

Choose $r = n^{\frac{1}{k+1}}$. Then

$$\begin{aligned}r^{-k}n + kr &= \left(n^{\frac{1}{k+1}}\right)^{-k}n + kn^{\frac{1}{k+1}} \\ &= n^{1-\frac{k}{k+1}} + kn^{\frac{1}{k+1}} \\ &= (k+1)n^{\frac{1}{k+1}}.\end{aligned}$$

Choosing $k = \Theta(\log n)$ gives a logarithmic running time.

7.6 Skip Lists

How to do insert and delete?

The cost of insert and delete is proportional to the number of elements in the list. Insert or delete may require a lot of reorganization.

Use randomization instead!

7.6 Skip Lists

How to do insert and delete?

- ▶ If we want that in L_i we always skip over roughly the same number of elements in L_{i-1} an insert or delete may require a lot of re-organisation.

Use randomization instead!

7.6 Skip Lists

How to do insert and delete?

- ▶ If we want that in L_i we always skip over roughly the same number of elements in L_{i-1} an insert or delete may require a lot of re-organisation.

Use randomization instead!

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

Delete:

You get all predecessors via backward pointers.

Delete x in all lists it actually appears in.

The time for both operations is dominated by the search time.

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

Delete:

You get all predecessors via backward pointers.

Delete x in all lists it actually appears in.

The time for both operations is dominated by the search time.

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

Delete:

▶ You get all predecessors via backward pointers.

▶ Delete x in all lists it actually appears in.

The time for both operations is dominated by the search time.

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

Delete:

Find all predecessors of the node to be deleted.

Remove it from all lists it actually appears in.

The time for both operations is dominated by the search time.

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

Delete:

- ▶ You get all predecessors via backward pointers.
- ▶ Delete x in all lists it actually appears in.

The time for both operations is dominated by the search time.

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

Delete:

- ▶ You get all predecessors via backward pointers.
- ▶ Delete x in all lists it actually appears in.

The time for both operations is dominated by the search time.

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

Delete:

- ▶ You get all predecessors via backward pointers.
- ▶ Delete x in all lists it actually appears in.

The time for both operations is dominated by the search time.

7.6 Skip Lists

Insert:

- ▶ A search operation gives you the insert position for element x in every list.
- ▶ Flip a coin until it shows head, and record the number $t \in \{1, 2, \dots\}$ of trials needed.
- ▶ Insert x into lists L_0, \dots, L_{t-1} .

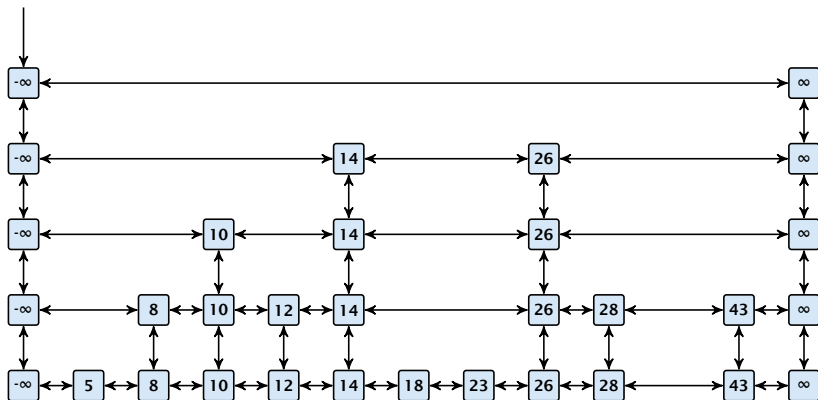
Delete:

- ▶ You get all predecessors via backward pointers.
- ▶ Delete x in all lists it actually appears in.

The time for both operations is dominated by the search time.

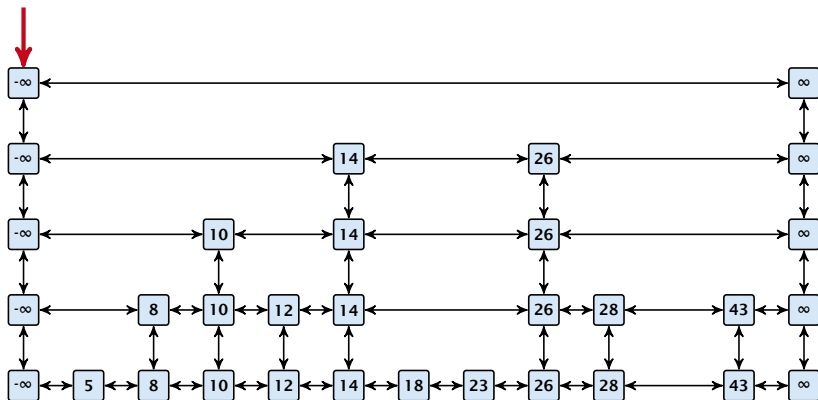
7.6 Skip Lists

Insert (35):



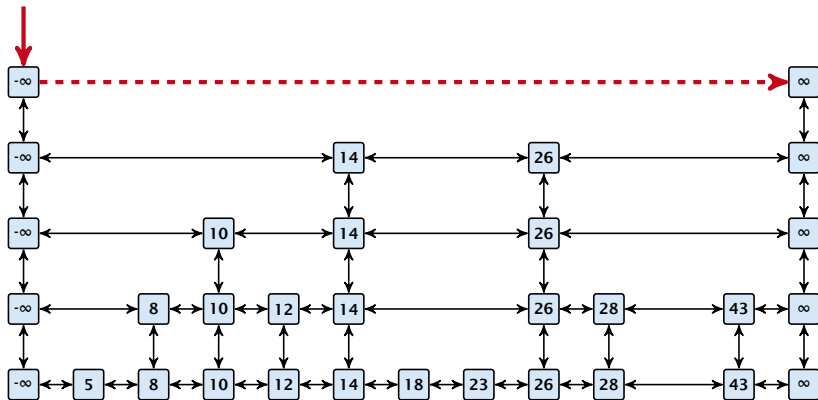
7.6 Skip Lists

Insert (35):



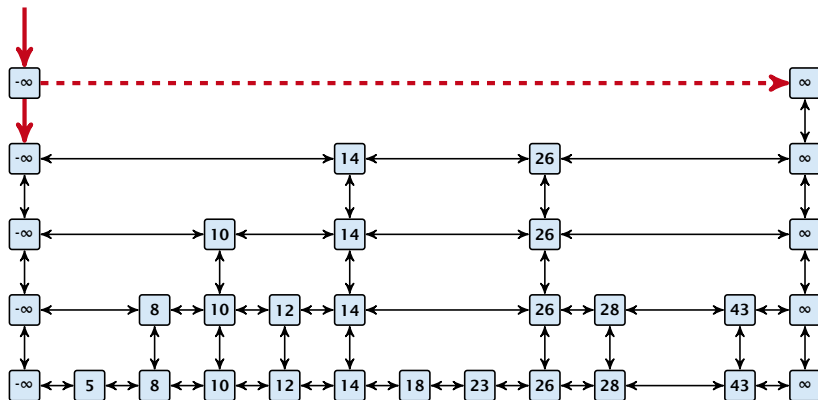
7.6 Skip Lists

Insert (35):



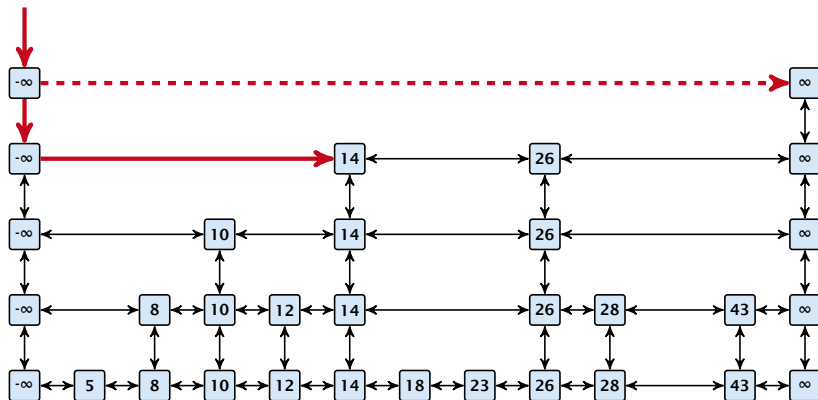
7.6 Skip Lists

Insert (35):



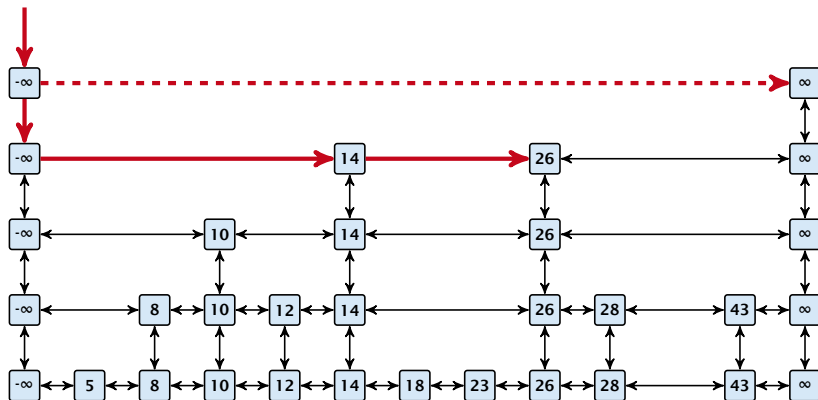
7.6 Skip Lists

Insert (35):



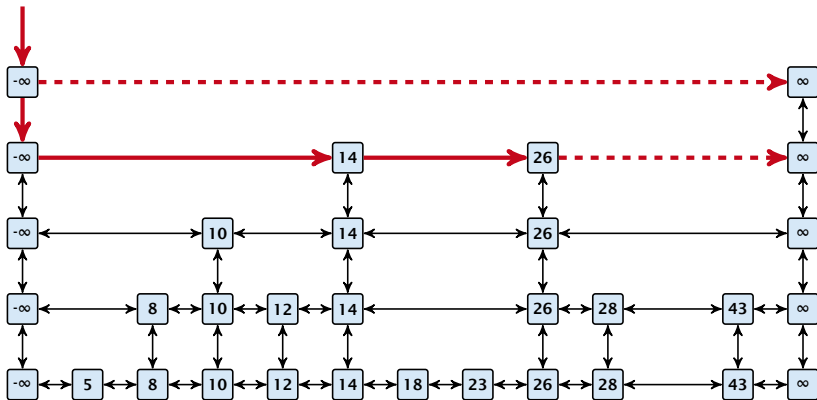
7.6 Skip Lists

Insert (35):



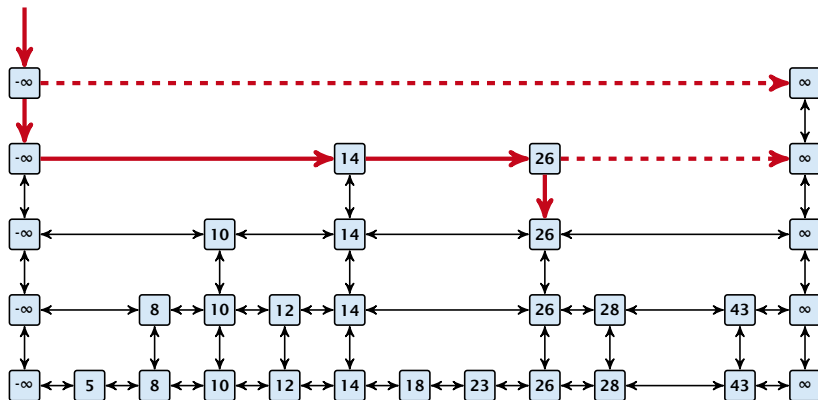
7.6 Skip Lists

Insert (35):



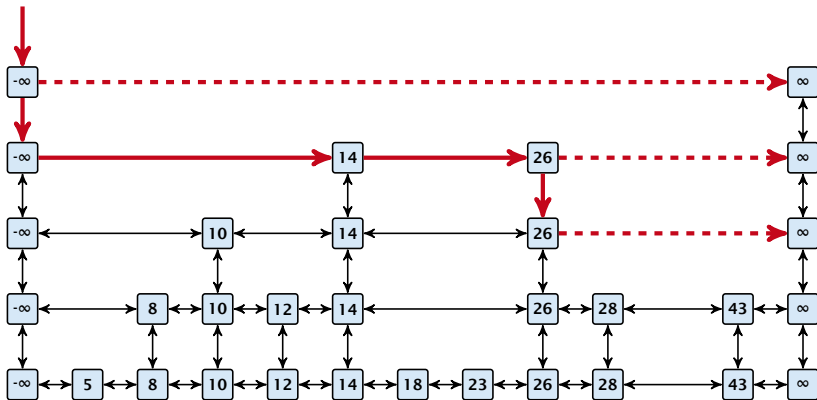
7.6 Skip Lists

Insert (35):



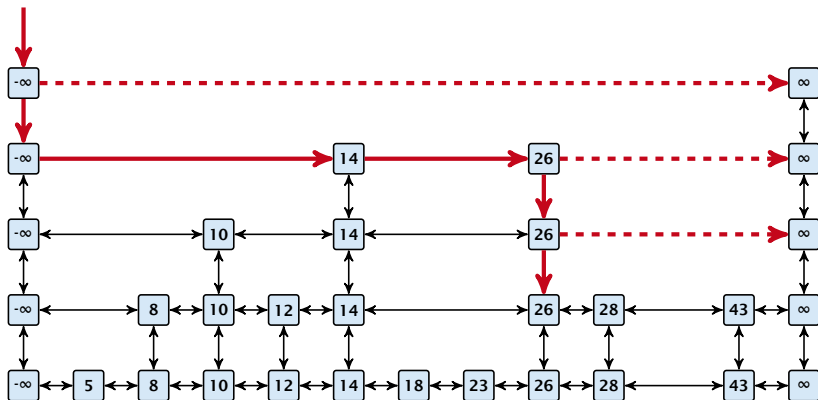
7.6 Skip Lists

Insert (35):



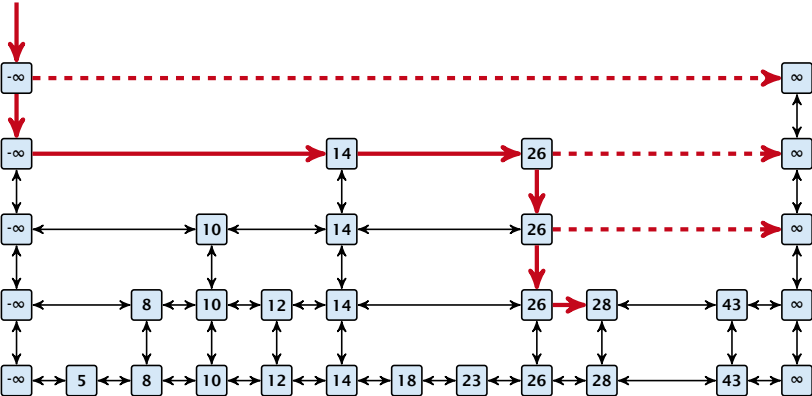
7.6 Skip Lists

Insert (35):



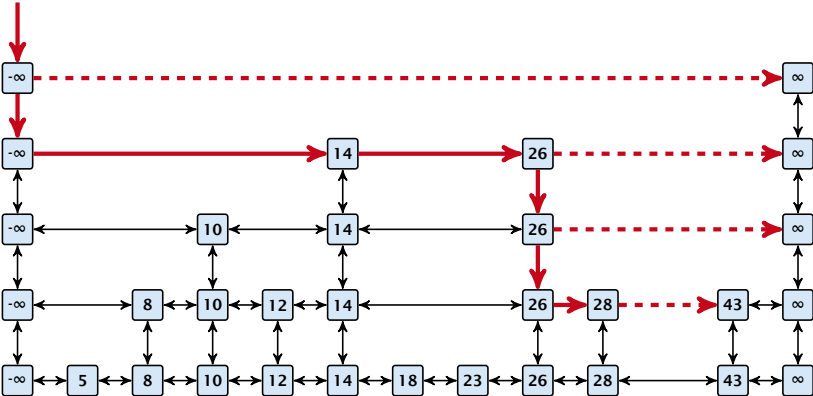
7.6 Skip Lists

Insert (35):



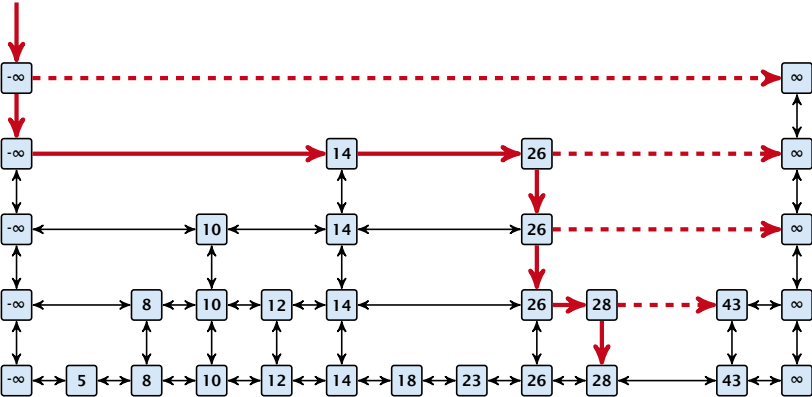
7.6 Skip Lists

Insert (35):



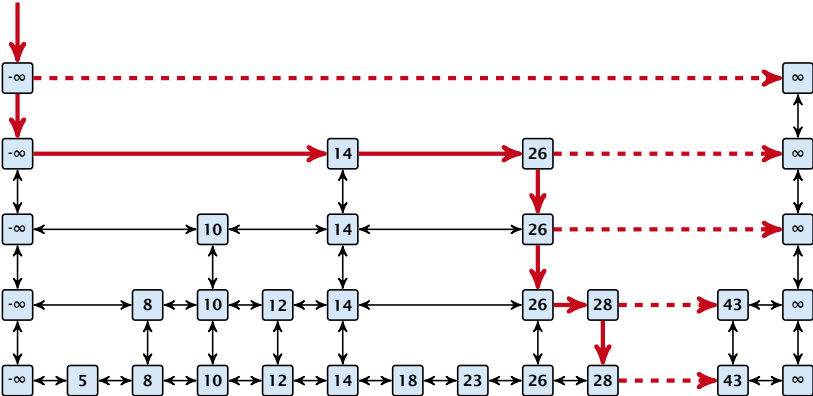
7.6 Skip Lists

Insert (35):



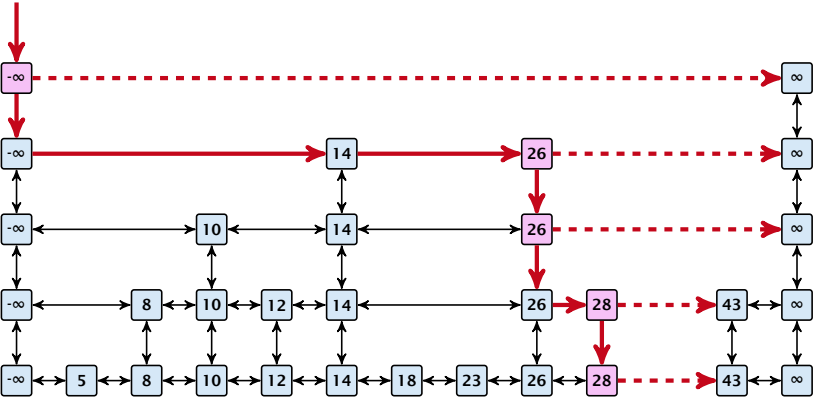
7.6 Skip Lists

Insert (35):



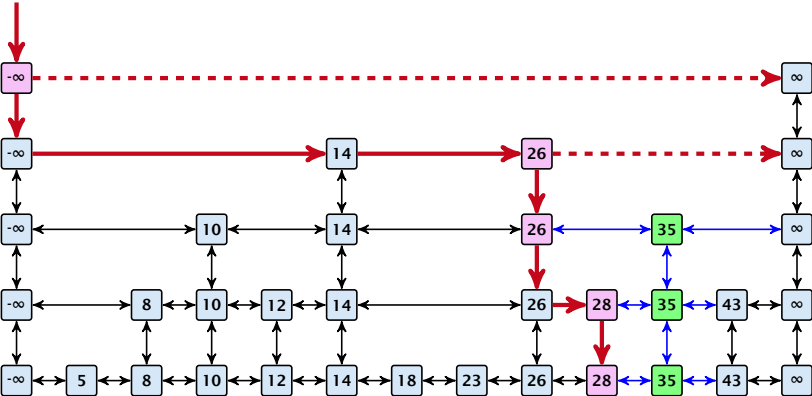
7.6 Skip Lists

Insert (35):



7.6 Skip Lists

Insert (35):



High Probability

Definition 1 (High Probability)

We say a **randomized** algorithm has running time $\mathcal{O}(\log n)$ with **high probability** if for any constant α the running time is at most $\mathcal{O}(\log n)$ with probability at least $1 - \frac{1}{n^\alpha}$.

Here the \mathcal{O} -notation hides a constant that may depend on α .

High Probability

Definition 1 (High Probability)

We say a **randomized** algorithm has running time $\mathcal{O}(\log n)$ with **high probability** if for any constant α the running time is at most $\mathcal{O}(\log n)$ with probability at least $1 - \frac{1}{n^\alpha}$.

Here the \mathcal{O} -notation hides a constant that may depend on α .

High Probability

Suppose there are a **polynomially** many events E_1, E_2, \dots, E_ℓ , $\ell = n^c$ each holding with high probability (e.g. E_i may be the event that the i -th search in a skip list takes time at most $\mathcal{O}(\log n)$).

High Probability

Suppose there are a **polynomially** many events E_1, E_2, \dots, E_ℓ , $\ell = n^c$ each holding with high probability (e.g. E_i may be the event that the i -th search in a skip list takes time at most $\mathcal{O}(\log n)$).

Then the probability that all E_i hold is at least

$$\Pr[E_1 \wedge \dots \wedge E_\ell]$$

High Probability

Suppose there are a **polynomially** many events E_1, E_2, \dots, E_ℓ , $\ell = n^c$ each holding with high probability (e.g. E_i may be the event that the i -th search in a skip list takes time at most $\mathcal{O}(\log n)$).

Then the probability that all E_i hold is at least

$$\Pr[E_1 \wedge \dots \wedge E_\ell] = 1 - \Pr[\bar{E}_1 \vee \dots \vee \bar{E}_\ell]$$

High Probability

Suppose there are a **polynomially** many events E_1, E_2, \dots, E_ℓ , $\ell = n^c$ each holding with high probability (e.g. E_i may be the event that the i -th search in a skip list takes time at most $\mathcal{O}(\log n)$).

Then the probability that all E_i hold is at least

$$\begin{aligned}\Pr[E_1 \wedge \dots \wedge E_\ell] &= 1 - \Pr[\bar{E}_1 \vee \dots \vee \bar{E}_\ell] \\ &\geq 1 - n^c \cdot n^{-\alpha}\end{aligned}$$

High Probability

Suppose there are a **polynomially** many events E_1, E_2, \dots, E_ℓ , $\ell = n^c$ each holding with high probability (e.g. E_i may be the event that the i -th search in a skip list takes time at most $\mathcal{O}(\log n)$).

Then the probability that all E_i hold is at least

$$\begin{aligned}\Pr[E_1 \wedge \dots \wedge E_\ell] &= 1 - \Pr[\bar{E}_1 \vee \dots \vee \bar{E}_\ell] \\ &\geq 1 - n^c \cdot n^{-\alpha} \\ &= 1 - n^{c-\alpha} .\end{aligned}$$

High Probability

Suppose there are a **polynomially** many events E_1, E_2, \dots, E_ℓ , $\ell = n^c$ each holding with high probability (e.g. E_i may be the event that the i -th search in a skip list takes time at most $\mathcal{O}(\log n)$).

Then the probability that all E_i hold is at least

$$\begin{aligned}\Pr[E_1 \wedge \dots \wedge E_\ell] &= 1 - \Pr[\bar{E}_1 \vee \dots \vee \bar{E}_\ell] \\ &\geq 1 - n^c \cdot n^{-\alpha} \\ &= 1 - n^{c-\alpha} .\end{aligned}$$

This means $\Pr[E_1 \wedge \dots \wedge E_\ell]$ holds with high probability.

7.6 Skip Lists

Lemma 2

A search (and, hence, also insert and delete) in a skip list with n elements takes time $\mathcal{O}(\log n)$ with high probability (w. h. p.).

7.6 Skip Lists

Backward analysis:



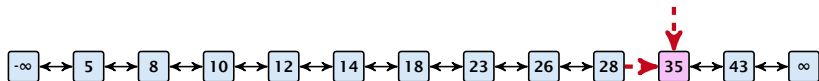
7.6 Skip Lists

Backward analysis:



7.6 Skip Lists

Backward analysis:



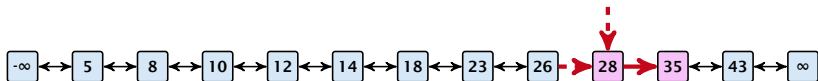
7.6 Skip Lists

Backward analysis:



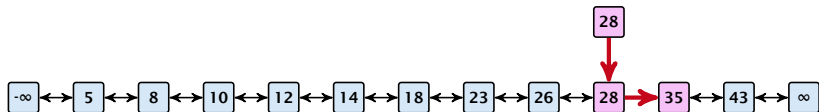
7.6 Skip Lists

Backward analysis:



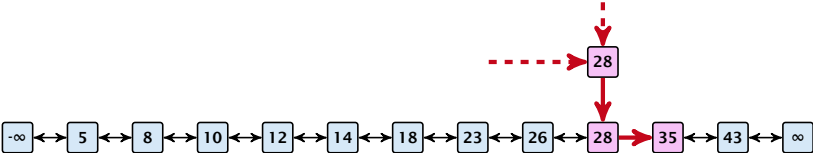
7.6 Skip Lists

Backward analysis:



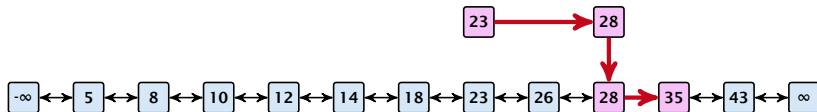
7.6 Skip Lists

Backward analysis:



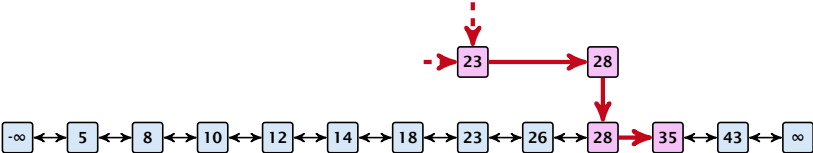
7.6 Skip Lists

Backward analysis:



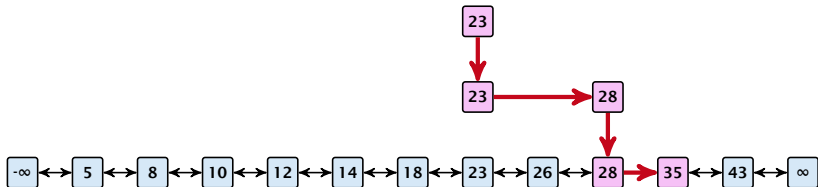
7.6 Skip Lists

Backward analysis:



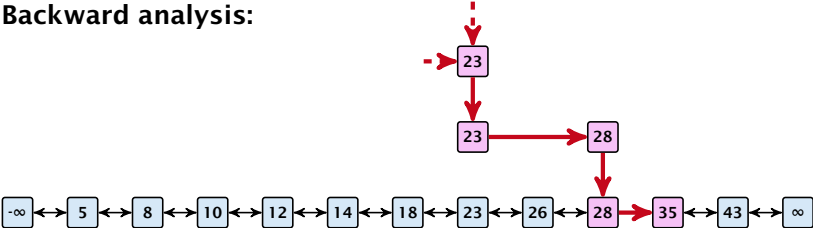
7.6 Skip Lists

Backward analysis:



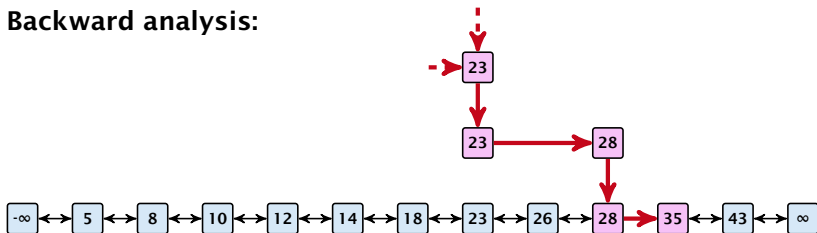
7.6 Skip Lists

Backward analysis:



7.6 Skip Lists

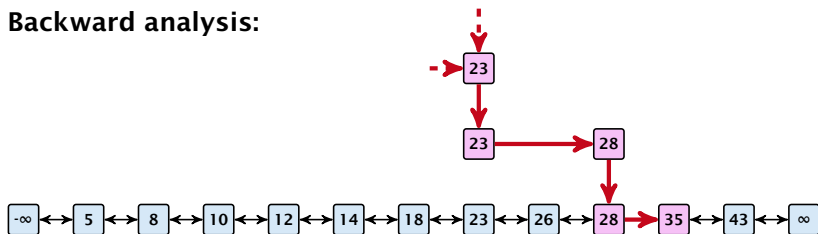
Backward analysis:



At each point the path goes up with probability $1/2$ and left with probability $1/2$.

7.6 Skip Lists

Backward analysis:



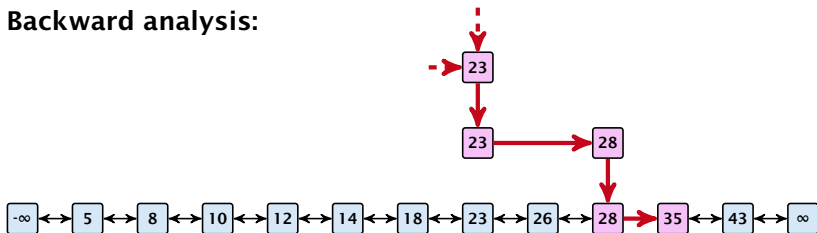
At each point the path goes up with probability $1/2$ and left with probability $1/2$.

We show that w.h.p:

- ▶ A “long” search path must also go very high.

7.6 Skip Lists

Backward analysis:



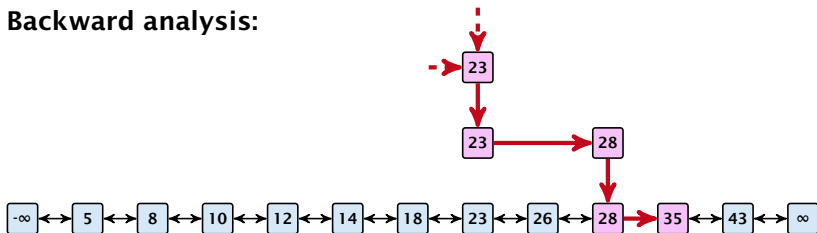
At each point the path goes up with probability $1/2$ and left with probability $1/2$.

We show that w.h.p:

- ▶ A “long” search path must also go very high.
- ▶ There are no elements in high lists.

7.6 Skip Lists

Backward analysis:



At each point the path goes up with probability $1/2$ and left with probability $1/2$.

We show that w.h.p.:

- ▶ A “long” search path must also go very high.
- ▶ There are no elements in high lists.

From this it follows that w.h.p. there are no long paths.

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1} \geq \left(\frac{n}{k}\right)^k$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1} \geq \left(\frac{n}{k}\right)^k$$

$$\binom{n}{k}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1} \geq \left(\frac{n}{k}\right)^k$$

$$\binom{n}{k} = \frac{n \cdot \dots \cdot (n-k+1)}{k!}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1} \geq \left(\frac{n}{k}\right)^k$$

$$\binom{n}{k} = \frac{n \cdot \dots \cdot (n-k+1)}{k!} \leq \frac{n^k}{k!}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1} \geq \left(\frac{n}{k}\right)^k$$

$$\binom{n}{k} = \frac{n \cdot \dots \cdot (n-k+1)}{k!} \leq \frac{n^k}{k!} = \frac{n^k \cdot k^k}{k^k \cdot k!}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1} \geq \left(\frac{n}{k}\right)^k$$

$$\begin{aligned} \binom{n}{k} &= \frac{n \cdot \dots \cdot (n-k+1)}{k!} \leq \frac{n^k}{k!} = \frac{n^k \cdot k^k}{k^k \cdot k!} \\ &= \left(\frac{n}{k}\right)^k \cdot \frac{k^k}{k!} \end{aligned}$$

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n \cdot \dots \cdot (n-k+1)}{k \cdot \dots \cdot 1} \geq \left(\frac{n}{k}\right)^k$$

$$\begin{aligned} \binom{n}{k} &= \frac{n \cdot \dots \cdot (n-k+1)}{k!} \leq \frac{n^k}{k!} = \frac{n^k \cdot k^k}{k^k \cdot k!} \\ &= \left(\frac{n}{k}\right)^k \cdot \frac{k^k}{k!} \leq \left(\frac{en}{k}\right)^k \end{aligned}$$

7.6 Skip Lists

7.6 Skip Lists

Let $E_{z,k}$ denote the event that a search path is of length z (number of edges) but does not visit a list above L_k .

7.6 Skip Lists

Let $E_{z,k}$ denote the event that a search path is of length z (number of edges) but does not visit a list above L_k .

In particular, this means that during the construction in the backward analysis we see at most k heads (i.e., coin flips that tell you to go up) in z trials.

7.6 Skip Lists

$$\Pr[E_{z,k}]$$

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

7.6 Skip Lists

$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$

$$\leq \binom{z}{k} 2^{-(z-k)}$$

7.6 Skip Lists

$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)}$$

7.6 Skip Lists

$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

7.6 Skip Lists

$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

$$\leq \left(\frac{2ez}{k}\right)^k 2^{-\beta k} \cdot n^{-\gamma\alpha}$$

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

$$\leq \left(\frac{2ez}{k}\right)^k 2^{-\beta k} \cdot n^{-\gamma\alpha} \leq \left(\frac{2ez}{2^\beta k}\right)^k \cdot n^{-\alpha}$$

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

$$\leq \left(\frac{2ez}{k}\right)^k 2^{-\beta k} \cdot n^{-\gamma\alpha} \leq \left(\frac{2ez}{2^\beta k}\right)^k \cdot n^{-\alpha}$$

$$\leq \left(\frac{2e(\beta + \alpha)}{2^\beta}\right)^k n^{-\alpha}$$

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

$$\leq \left(\frac{2ez}{k}\right)^k 2^{-\beta k} \cdot n^{-\gamma\alpha} \leq \left(\frac{2ez}{2^\beta k}\right)^k \cdot n^{-\alpha}$$

$$\leq \left(\frac{2e(\beta + \alpha)}{2^\beta}\right)^k n^{-\alpha}$$

now choosing $\beta = 6\alpha$ gives

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

$$\leq \left(\frac{2ez}{k}\right)^k 2^{-\beta k} \cdot n^{-\gamma\alpha} \leq \left(\frac{2ez}{2^\beta k}\right)^k \cdot n^{-\alpha}$$

$$\leq \left(\frac{2e(\beta + \alpha)}{2^\beta}\right)^k n^{-\alpha}$$

now choosing $\beta = 6\alpha$ gives

$$\leq \left(\frac{42\alpha}{64\alpha}\right)^k n^{-\alpha}$$

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

$$\leq \left(\frac{2ez}{k}\right)^k 2^{-\beta k} \cdot n^{-\gamma\alpha} \leq \left(\frac{2ez}{2^\beta k}\right)^k \cdot n^{-\alpha}$$

$$\leq \left(\frac{2e(\beta + \alpha)}{2^\beta}\right)^k n^{-\alpha}$$

now choosing $\beta = 6\alpha$ gives

$$\leq \left(\frac{42\alpha}{64\alpha}\right)^k n^{-\alpha} \leq n^{-\alpha}$$

7.6 Skip Lists

$$\Pr[E_{z,k}] \leq \Pr[\text{at most } k \text{ heads in } z \text{ trials}]$$

$$\leq \binom{z}{k} 2^{-(z-k)} \leq \left(\frac{ez}{k}\right)^k 2^{-(z-k)} \leq \left(\frac{2ez}{k}\right)^k 2^{-z}$$

choosing $k = \gamma \log n$ with $\gamma \geq 1$ and $z = (\beta + \alpha)\gamma \log n$

$$\leq \left(\frac{2ez}{k}\right)^k 2^{-\beta k} \cdot n^{-\gamma\alpha} \leq \left(\frac{2ez}{2^\beta k}\right)^k \cdot n^{-\alpha}$$

$$\leq \left(\frac{2e(\beta + \alpha)}{2^\beta}\right)^k n^{-\alpha}$$

now choosing $\beta = 6\alpha$ gives

$$\leq \left(\frac{42\alpha}{64\alpha}\right)^k n^{-\alpha} \leq n^{-\alpha}$$

for $\alpha \geq 1$.

7.6 Skip Lists

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

Let A_{k+1} denote the event that the list L_{k+1} is non-empty. Then

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

Let A_{k+1} denote the event that the list L_{k+1} is non-empty. Then

$$\Pr[A_{k+1}] \leq n2^{-(k+1)} \leq n^{-(\gamma-1)} .$$

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

Let A_{k+1} denote the event that the list L_{k+1} is non-empty. Then

$$\Pr[A_{k+1}] \leq n2^{-(k+1)} \leq n^{-(\gamma-1)} .$$

For the search to take at least $z = 7\alpha\gamma \log n$ steps either the event $E_{z,k}$ or the event A_{k+1} must hold.

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

Let A_{k+1} denote the event that the list L_{k+1} is non-empty. Then

$$\Pr[A_{k+1}] \leq n2^{-(k+1)} \leq n^{-(\gamma-1)} .$$

For the search to take at least $z = 7\alpha\gamma \log n$ steps either the event $E_{z,k}$ or the event A_{k+1} must hold.

Hence,

$$\Pr[\text{search requires } z \text{ steps}]$$

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

Let A_{k+1} denote the event that the list L_{k+1} is non-empty. Then

$$\Pr[A_{k+1}] \leq n2^{-(k+1)} \leq n^{-(\gamma-1)} .$$

For the search to take at least $z = 7\alpha\gamma \log n$ steps either the event $E_{z,k}$ or the event A_{k+1} must hold.

Hence,

$$\Pr[\text{search requires } z \text{ steps}] \leq \Pr[E_{z,k}] + \Pr[A_{k+1}]$$

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

Let A_{k+1} denote the event that the list L_{k+1} is non-empty. Then

$$\Pr[A_{k+1}] \leq n2^{-(k+1)} \leq n^{-(\gamma-1)} .$$

For the search to take at least $z = 7\alpha\gamma \log n$ steps either the event $E_{z,k}$ or the event A_{k+1} must hold.

Hence,

$$\begin{aligned} \Pr[\text{search requires } z \text{ steps}] &\leq \Pr[E_{z,k}] + \Pr[A_{k+1}] \\ &\leq n^{-\alpha} + n^{-(\gamma-1)} \end{aligned}$$

7.6 Skip Lists

So far we fixed $k = \gamma \log n$, $\gamma \geq 1$, and $z = 7\alpha\gamma \log n$, $\alpha \geq 1$.

This means that a search path of length $\Omega(\log n)$ visits a list on a level $\Omega(\log n)$, w.h.p.

Let A_{k+1} denote the event that the list L_{k+1} is non-empty. Then

$$\Pr[A_{k+1}] \leq n2^{-(k+1)} \leq n^{-(\gamma-1)} .$$

For the search to take at least $z = 7\alpha\gamma \log n$ steps either the event $E_{z,k}$ or the event A_{k+1} must hold.

Hence,

$$\begin{aligned} \Pr[\text{search requires } z \text{ steps}] &\leq \Pr[E_{z,k}] + \Pr[A_{k+1}] \\ &\leq n^{-\alpha} + n^{-(\gamma-1)} \end{aligned}$$

This means, the search requires at most z steps, w. h. p.