Analysis of Fully Distributed Splitting and Naming Probabilistic Procedures and Applications

Y. Métivier, J.M. Robson, A. Zemmari

LaBRI, University of Bordeaux

SIROCCO 2013
July 1-3, 2013, Ischia, Italy

Introduction

Probabilistic distributed (splitting + naming) to solve:

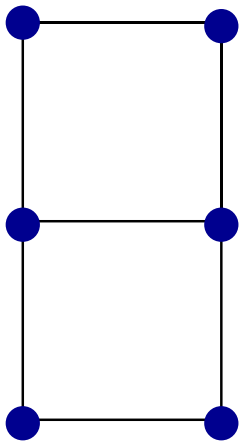
- Election
- Counting
- Spanning trees.

Introduction

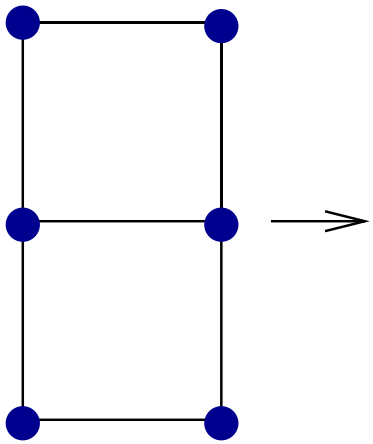
Probabilistic distributed (splitting + naming) to solve:

- Election
- Counting
- Spanning trees.

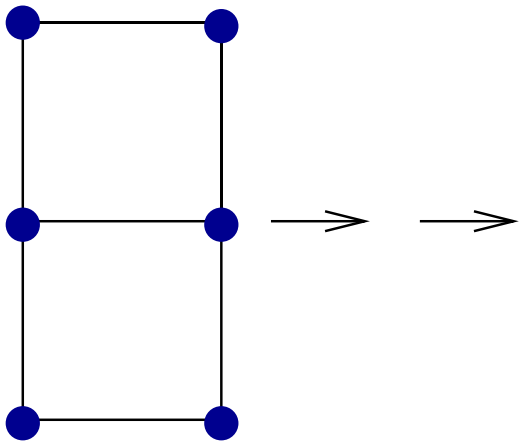
The Election Problem



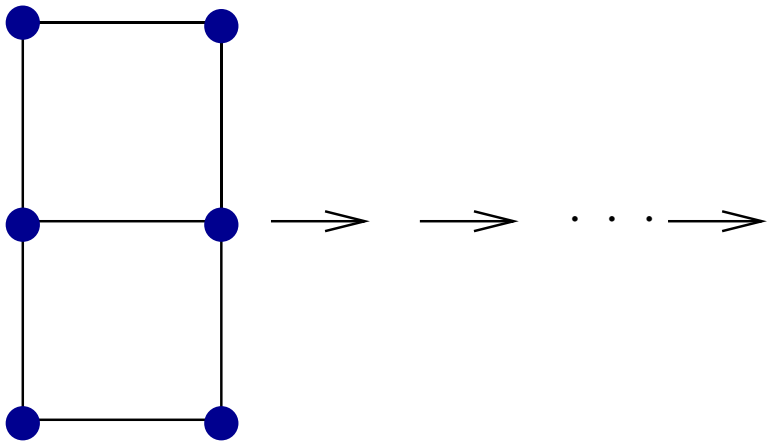
The Election Problem



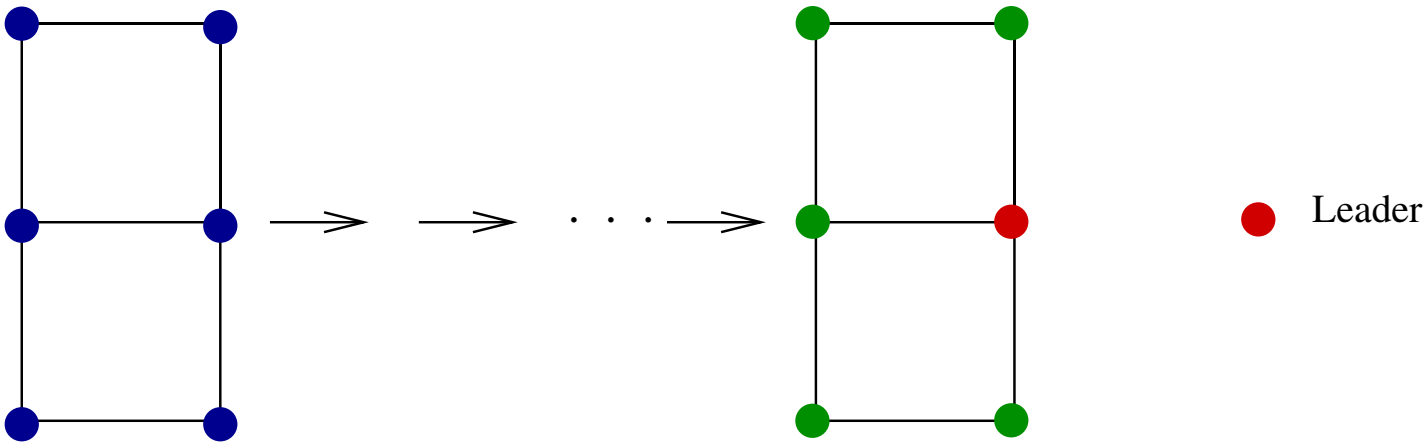
The Election Problem



The Election Problem



The Election Problem



The Election Problem

Definition [LeLann,Tel]

An election algorithm is an algorithm that satisfies the following properties

1. Each process has the same local algorithm.
2. The algorithm is decentralized.
3. The algorithm reaches a terminal configuration in which there is exactly one process in the state leader and all the other processes are in the state lost.

The Model

- Anonymous networks.
- Point to point communication using messages passing.
- No global knowledge (size, diameter, ...)
- Nodes can distinguish between neighbors using port numbers.

$\Rightarrow$ the network is modeled by a graph $G = (V, E)$.
n denotes the size of the graph and m the number of edges.

The Model

- Anonymous networks.
- Point to point communication using messages passing.
- No global knowledge (size, diameter, ...)
- Nodes can distinguish between neighbors using port numbers.

$\Rightarrow$ the network is modeled by a graph $G = (V, E)$.
n denotes the size of the graph and m the number of edges.

The Model

- Anonymous networks.
- Point to point communication using messages passing.
- No global knowledge (size, diameter, ...)
- Nodes can distinguish between neighbors using port numbers.

$\Rightarrow$ the network is modeled by a graph $G = (V, E)$.
n denotes the size of the graph and m the number of edges.

The Model

- Anonymous networks.
- Point to point communication using messages passing.
- No global knowledge (size, diameter, ...)
- Nodes can distinguish between neighbors using port numbers.

$\Rightarrow$ the network is modeled by a graph $G = (V, E)$.
n denotes the size of the graph and m the number of edges.

The Model

- Anonymous networks.
- Point to point communication using messages passing.
- No global knowledge (size, diameter, ...)
- Nodes can distinguish between neighbors using port numbers.

$\Rightarrow$ the network is modeled by a graph $G = (V, E)$.
n denotes the size of the graph and m the number of edges.

The Model

- Anonymous networks.
- Point to point communication using messages passing.
- No global knowledge (size, diameter, ...)
- Nodes can distinguish between neighbors using port numbers.

$\Rightarrow$ the network is modeled by a graph $G = (V, E)$.
n denotes the size of the graph and m the number of edges.

The Model

Definitions [Tel]:

- An algorithm *process terminates* if in every execution, all processors reach a special *halting state*.
- An algorithm *message terminates* if in every execution, the network reaches a quiescent state where there are no pending messages on the links.

Notation:

- An algorithm is correct with high probability (w.h.p. for short) if it is correct with probability $1 - o(1/n)$.
- An algorithm is correct with very high probability (w.v.h.p. for short) if it is correct with probability $1 - o(1/n^c)$ for any $c \geq 1$.

The Model

Definitions [Tel]:

- An algorithm *process terminates* if in every execution, all processors reach a special *halting state*.
- An algorithm *message terminates* if in every execution, the network reaches a quiescent state where there are no pending messages on the links.

Notation:

- An algorithm is correct with high probability (w.h.p. for short) if it is correct with probability $1 - o(1/n)$.
- An algorithm is correct with very high probability (w.v.h.p. for short) if it is correct with probability $1 - o(1/n^c)$ for any $c \geq 1$.

The Model

Definitions [Tel]:

- An algorithm *process terminates* if in every execution, all processors reach a special *halting state*.
- An algorithm *message terminates* if in every execution, the network reaches a quiescent state where there are no pending messages on the links.

Notation:

- An algorithm is correct with high probability (w.h.p. for short) if it is correct with probability $1 - o(1/n)$.
- An algorithm is correct with very high probability (w.v.h.p. for short) if it is correct with probability $1 - o(1/n^c)$ for any $c \geq 1$.

Related Works

Negative results:

- No deterministic algorithm can elect in anonymous networks (even if the size of the network is known) [Angluin].
- With no knowledge on the (anonymous) network, there exists no Las Vegas election algorithm [Itai and Rodeh].
- There exists no process terminating election algorithm that is correct with probability $p > 0$ [Itai and Rodeh].

Related Works

Negative results:

- No deterministic algorithm can elect in anonymous networks (even if the size of the network is known) [Angluin].
- With no knowledge on the (anonymous) network, there exists no Las Vegas election algorithm [Itai and Rodeh].
- There exists no process terminating election algorithm that is correct with probability $p > 0$ [Itai and Rodeh].

Related Works

Negative results:

- No deterministic algorithm can elect in anonymous networks (even if the size of the network is known) [Angluin].
- With no knowledge on the (anonymous) network, there exists no Las Vegas election algorithm [Itai and Rodeh].
- There exists no process terminating election algorithm that is correct with probability $p > 0$ [Itai and Rodeh].

Related Works

Negative results:

- No deterministic algorithm can elect in anonymous networks (even if the size of the network is known) [Angluin].
- With no knowledge on the (anonymous) network, there exists no Las Vegas election algorithm [Itai and Rodeh].
- There exists no process terminating election algorithm that is correct with probability $p > 0$ [Itai and Rodeh].

Related Works

- Symmetry breaking in distributed networks, A. Itai and M. Rodeh, Inf. Comput., 1990.
- Elections in Anonymous Networks, Y. Afek and Y. Matias, Inf. Comput., 1994.
- Calling Names on Nameless Networks, B. Schieber and M. Snir, Inf. Comput., 1994.
- On the Complexity of Universal Leader Election, S. Kutten, G. Pandurangan, D. Peleg, P. Robinson, and A. Trehan, PODC 2013.

A Splitting and Naming Probabilistic Procedure

Algorithm 1 Procedure Splitting-Naming-whp(v).

1: $t_v := 0$
2: **repeat**
3: draw uniformly at random a bit $b(v)$;
4: $t_v := t_v + 1$
5: **until** $b(v) = 1$
6: choose uniformly at random a number id_v in the set $\{0, \cdots, 2^{t_v + 3\log_2(t_v)} - 1\}$;

$\Rightarrow$ the label of each vertex v is (t_v, id_v).

A Splitting and Naming Probabilistic Procedure

Algorithm 2 Procedure Splitting-Naming-whp(v).

1: $t_v := 0$
2: **repeat**
3: draw uniformly at random a bit $b(v)$;
4: $t_v := t_v + 1$
5: **until** $b(v) = 1$
6: choose uniformly at random a number id_v in the set $\{0, \cdots, 2^{t_v + 3\log_2(t_v)} - 1\}$;

$\Rightarrow$ the label of each vertex v is (t_v, id_v).

A Splitting and Naming Probabilistic Procedure

Algorithm 3 Procedure Splitting-Naming-whp(v).

1: $t_v := 0$
2: **repeat**
3: draw uniformly at random a bit $b(v)$;
4: $t_v := t_v + 1$
5: **until** $b(v) = 1$
6: choose uniformly at random a number id_v in the set $\{0, \cdots, 2^{t_v + 3\log_2(t_v)} - 1\}$;

$\Rightarrow$ the label of each vertex v is (t_v, id_v).

A Splitting and Naming Probabilistic Procedure

Algorithm 4 Procedure Splitting-Naming-whp(v).

1: $t_v := 0$
2: **repeat**
3: draw uniformly at random a bit $b(v)$;
4: $t_v := t_v + 1$
5: **until** $b(v) = 1$
6: choose uniformly at random a number id_v in the set $\{0, \cdots, 2^{t_v + 3\log_2(t_v)} - 1\}$;

$\Rightarrow$ the label of each vertex v is (t_v, id_v).

A Splitting and Naming Probabilistic Procedure

We define the order, denoted $<$, on couples by:
$(t_v, id_v) < (t'_v, id'_v)$ if:

- either $t_v < t'_v$
- or $t_v = t'_v$ and $id_v < id'_v$.

We analyse Procedure Splitting-Naming-whp(v), and more precisely:

1. the lifetime of each vertex,
2. the number of vertices that share the maximal lifetime,
3. the number of vertices that share the maximal lifetime and the same maximal number,
4. the size of the labels.

Analysis of the Procedure

The maximal number of bits drawn by each vertex:

Theorem 1.

Let $G = (V, E)$ be a graph with $n > 0$ vertices. Let T denote the maximal value of the set $\{t_v | v \in V\}$.

1. $T < 2 \log_2 n + \log^* n$ w.h.p.
2. $T > \log_2 n - \log_2 (2 \log n)$ w.h.p.

Corollary 2.

The expected value of T is equal to $\Theta(\log n)$.
The labels are of size at most $2 \log_2 n + O(\log_2 \log_2 n)$ w.h.p.
The expected value of the size of the labels is $O(\log n)$.

Analysis of the Procedure

The number of vertices that share the maximal lifetime:

Theorem 3.

The number of vertices which have the same maximum lifetime is, with high probability, at most $2 \log_2 n$.

Analysis of the Procedure

The number of vertices that share the maximal lifetime and the same maximal number:

Lemma 4.

With high probability, there exists a unique vertex v with a couple (t_v, id_v) such that for any $w \in V \setminus \{v\}$, $t_v \geq t_w$ and $id_v > id_w$.

Braodcasting the Maximum

Algorithm 5 Broadcasting-Max.

I : {If $label_v$ is not defined}

choose at random $label_v$ from L;

$max_v := label_v$;

send $< (label_v) >$ to all neighbours

B : {A Message ($label$) has arrived at v through port p}

If $label_v$ is not defined

choose at random $label_v$ from L;

$max_v := label_v$;

send $< label_v >$ to all neighbours

If $label > max_v$

$max_v := label$;

send $< (label) >$ to all neighbours except through port p

Message Complexity

Proposition.

Let G be a graph and let L be a totally ordered set of labels. Each vertex v of G chooses a label from L; all the vertices use the same distribution to choose a label. For each run of Algorithm Broadcasting-Max the number of messages sent by each vertex of G is w.h.p. $O(\log n)$.

Corollary.

Procedure Broadcasting-Max has a message complexity equal to $O(m \log n)$ w.h.p.

Message Complexity

Proposition.

Let G be a graph and let L be a totally ordered set of labels. Each vertex v of G chooses a label from L; all the vertices use the same distribution to choose a label. For each run of Algorithm Broadcasting-Max the number of messages sent by each vertex of G is w.h.p. $O(\log n)$.

Corollary.

Procedure Broadcasting-Max has a message complexity equal to $O(m \log n)$ w.h.p.

MC Election Algorithm

Splitting-Naming-whp + Broadcasting-Max $\Rightarrow$ An election algorithm

- correct w.h.p.,
- using messages of size $O(\log n)$ w.h.p.,
- with a message complexity $O(m \log n)$.

The Second MC Algorithm

In Algorithm Elect

The label of each node v is a couple (t_v, id_v) where:
- t_v is the number of draws,
- id_v is a number chosen u.a.r. in the set $\{0, ..., 2^{t_v \log^* t_v} - 1\}$.

Analysis of the Algorithm

Lemma 5.

Let $G = (V, E)$ be a graph with $n > 0$ vertices. Let T denote the maximal value of the set $\{t_v | v \in V\}$. W.v.h.p., T satisfies:

$$\frac{1}{2} \log_2 n < T < (\log_2 n) \log^* n.$$

Theorem 6.

There exists a Monte Carlo election algorithm that succeeds w.v.h.p.

Proposition 7.

The size of messages is w.v.h.p. $O\left((\log n)(\log^* n)^2\right)$. Its expected value is $O\left((\log n)(\log^* n)\right)$.

Analysis of the Algorithm

Lemma 5.

Let $G = (V, E)$ be a graph with $n > 0$ vertices. Let T denote the maximal value of the set $\{t_v | v \in V\}$. W.v.h.p., T satisfies:

$$\frac{1}{2} \log_2 n < T < (\log_2 n) \log^* n.$$

Theorem 6.

There exists a Monte Carlo election algorithm that succeeds w.v.h.p.

Proposition 7.

The size of messages is w.v.h.p. $O\left((\log n)(\log^* n)^2\right)$. Its expected value is $O\left((\log n)(\log^* n)\right)$.

Analysis of the Algorithm

Lemma 5.

Let $G = (V, E)$ be a graph with $n > 0$ vertices. Let T denote the maximal value of the set $\{t_v | v \in V\}$. W.v.h.p., T satisfies:

$$\frac{1}{2} \log_2 n < T < (\log_2 n) \log^* n.$$

Theorem 6.

There exists a Monte Carlo election algorithm that succeeds w.v.h.p.

Proposition 7.

The size of messages is w.v.h.p. $O\left((\log n)(\log^* n)^2\right)$. Its expected value is $O\left((\log n)(\log^* n)\right)$.

Conclusion and Perspectives

We can use Splitting-Naming-whp to:

- compute a spanning tree,
- count the number of nodes in a cycle.

Can we use the splitting-naming procedure to improve the message complexity in networks with identifiers ?

Conclusion and Perspectives

Assume each vertex v aims to ensure an election correct with probability at least $1 - \varepsilon_v$, (for $\varepsilon_v > 0$).
$\Rightarrow$ node v acts as a set of $K_v = f(\varepsilon_v)$ nodes.

Thank You

Annex

Proof of Theorem 1.

After t rounds the expected number of vertices still alive is $\frac{n}{2^t}$. In particular after $2\log_2 n + \log^* n$ rounds it is $\frac{1}{n2^{\log^* n}}$ so that the probability that any vertex remains is less than $\frac{1}{n2^{\log^* n}}$. This proves the first claim.
On the other hand, for any $t > 0$, we have the following equality:

$$\mathbb{P}r\left(T > t\right) = \mathbb{P}r\left(\exists v \in V \text{ s.t. } t_v > t\right) = 1 - \left(1 - \frac{1}{2^t}\right)^n.$$

We also have the exponential approximation $(1-a)^n \sim e^{-an}$ as $n \to \infty$, thus: $\mathbb{P}r\left(T > t\right) \sim 1 - e^{-\frac{n}{2^t}}$, as $n \to \infty$.
Taking $t = \log_2 n - \log_2\left(2\log n\right)$, we obtain: $\mathbb{P}r\left(T > t\right) \geq 1 - o\left(\frac{1}{n}\right)$, this proves the second claim which ends the proof. □

Proof of Corollary 2.

At the end of each round, half of the vertices still active become inactive. Thus $\mathbb{E}(T)$, the expected value of T, is $O(\log n)$.
On the other hand, using the Markov inequality, we have:

$$\begin{aligned} \mathbb{E}(T) &\geq (\log_2 n - \log_2(2\log n)) \\ &\times \mathbb{P}r(T > \log_2 n - \log_2(2\log n)). \end{aligned} \tag{1}$$

Then, using the second claim of Theorem 16, this proves that $\mathbb{E}(T) = \Omega(\log n)$. Which ends the proof. □

Proof of Theorem 4.

We consider the probability C_t that at time t, there are still more than $2\log n$ vertices alive and that they all vanish at the next round. The sum of all C_t is the probability that there are more than $2\log_2 n$ vertices sharing the same maximum lifetime. We have:

$$\begin{aligned} C_t &= \mathbb{P}r\left[\text{alive vertices at time } t > 2\log_2 n\right] \\ &\times \mathbb{P}r\left[\text{all alive vertices finish } \middle|\ \text{there are } > 2\log_2 n\right] \\ &\leq \mathbb{P}r\left[\text{all alive vertices finish } \middle|\ \text{there are} > 2\log_2 n\right]. \end{aligned}$$

Then, using (**??**), we obtain:

$$C_t \leq 2^{-2\log_2 n} = n^{-2}. \tag{2}$$

On the other hand, we have:

$$\mathbb{P}r\left(X_T \geq 2\log_2 n\right) \leq \mathbb{P}r\left(T > 2\log_2 n\right) + \sum_{t \leq 2\log_2 n} C_t = o\left(\frac{1}{n}\right). \tag{3}$$

Which ends the proof. □

Proof of Lemma 5.

Let $S = \{v_1, \cdots, v_k\}$ be the set of vertices that share the maximum lifetime. By Claim *2.* of Theorem 16, w.h.p., any vertex v in S is such that $t_v > \log_2 n - \log_2(2 \log n)$. Thus, each vertex v in S will choose u.a.r. an identifier id_v from a set which contains w.h.p. the set: $\left\{0, \cdots, \frac{n}{2 \log n} \times \left(\log_2 \left(\frac{n}{2 \log n}\right)\right)^3 - 1\right\}$.

If we denote $f(n) = \frac{n}{2\log n} \times \left(\log_2\left(\frac{n}{2\log n}\right)\right)^3$, then, the probability for v to be the unique vertex with the highest identifier is given by:

$$\mathbb{P}r\left(id_v > id_w, \forall w \in S \setminus \{v\}\right) \geq \sum_{i=1}^{f(n)-1} \left(\frac{1}{f(n)}\right)^k (i-1)^{k-1} \sim$$

$\frac{1}{k}\left(1 - \frac{1}{f(n)}\right)^k$ as $n \to \infty$.

Thus, the probability that a unique vertex in S has the highest identifier is given by:

$$\mathbb{P}r\left(\exists v \text{ s.t. } id_v > id_w, \forall w \in S \setminus \{v\}\right) \sim \left(1 - \frac{1}{f(n)}\right)^k \text{ as } n \to \infty.$$

Now, by Theorem 17, we have $k < 2\log_2 n$, w.h.p., thus :

$$\begin{aligned}
&\mathbb{P}r\left(\exists v \text{ s.t. } id_v > id_w, \forall w \in S \setminus \{v\}\right) \\
&\geq \left(1 - \frac{1}{f(n)}\right)^{2\log_2 n} \\
&\sim e^{-2\frac{\log_2 n}{f(n)}} = 1 - o\left(\frac{1}{n}\right),
\end{aligned}$$

which ends the proof. □

Proof of Lemma 6.

Taking $t = \frac{1}{2}\log_2 n$ in (**??**), we obtain:

$$\begin{aligned} \mathbb{P}r\left(T > \tfrac{1}{2}\log_2 n\right) &\sim 1 - e^{-\sqrt{n}}, \quad \text{as } n \to \infty \\ &\sim 1 - o\left(\tfrac{1}{n^c}\right) \text{ for any } c \geq 1. \end{aligned}$$

On the other hand, after $(\log^* n)\log_2 n$ rounds, the expected number of vertices still alive is $\frac{n}{n^{\log^* n}}$ so that the probability that any vertex remains is less than $\frac{1}{n^{\log^* n - 1}}$ which is $o(\frac{1}{n^c})$ for any $c \geq 1$.

Proof of Proposition 8.

The first part is a direct consequence of Lemma 2 and of the choice of id_v in the set:

$$\left\{0, \cdots, 2^{T \log^* T} - 1\right\}.$$

For the second part, let $S(id_v)$ denote the size of id_v, and let $S_{max} = \max_{v \in V} S(id_v) = T \log^* T$.
Then:

$$\begin{aligned}
\mathbb{E}(S_{max}) &= \sum_{\substack{x \geq 1 \\ (\log n)(\log^* n)}} \mathbb{P}r(S_{max} \geq x) \\
&= \sum_{x=1} \mathbb{P}r(S_{max} \geq x) \\
&+ \sum_{x > (\log n)(\log^* n)} \mathbb{P}r(S_{max} \geq x),
\end{aligned}$$

Proof of Proposition 8.

yielding:

$$\mathbb{E}(S_{max}) \leq (\log n)(\log^* n) + \sum_{x \geq (\log n)(\log^*(\log n))} \mathbb{P}r(S_{max} \geq x). \quad (4)$$

Then the second part of the sum in expression (4) satisfies:

$$\sum_{x \geq (\log n)(\log^*(\log n))} \mathbb{P}r(S_{max} \geq x)$$

$$= \sum_{t \geq \log n} \sum_{y = t \log^* t}^{(t+1)\log^*(t+1)-1} \mathbb{P}r(S_{max} \geq y). \quad (5)$$

Proof of Proposition 8.

On the other hand, for any y in the interval $[t \log^* t, (t+1)\log^*(t+1) - 1]$:

$$\begin{aligned} \mathbb{P}r\left(S_{max} \geq y\right) &= \mathbb{P}r\left(S_{max} \geq t \log^* t\right) \\ &= \mathbb{P}r\left(T \geq t\right) \leq \frac{n}{2^t}, \end{aligned} \tag{6}$$

yielding:

$$\sum_{x \geq (\log n)(\log^*(\log n))} \mathbb{P}r\left(S_{max} \geq x\right)$$

$$\leq \sum_{t \geq \log n} \frac{n\left((t+1)\log^*(t+1) - t\log^* t\right)}{2^t}. \tag{7}$$

Since $\log^*(t+1) \leq \log^* t + 1$, this gives:

$$\sum_{x \geq (\log n)(\log^*(\log n))} \mathbb{P}r\left(S_{max} \geq x\right)$$

Proof of Proposition 8.

Hence:

$$\begin{aligned} \mathbb{E}(S_{max}) &\leq (\log n)(\log^* n) + O(\log n) \\ &= O((\log n)(\log^* n)), \end{aligned} \tag{9}$$

which ends the proof. □