Sublinear Random Access Generators for Preferential Attachment Graphs

Guy Even - Tel Aviv Univ.

Reut Levi, **Moti Medina** } MPI - SB

Adi Rosen - Paris Diderot

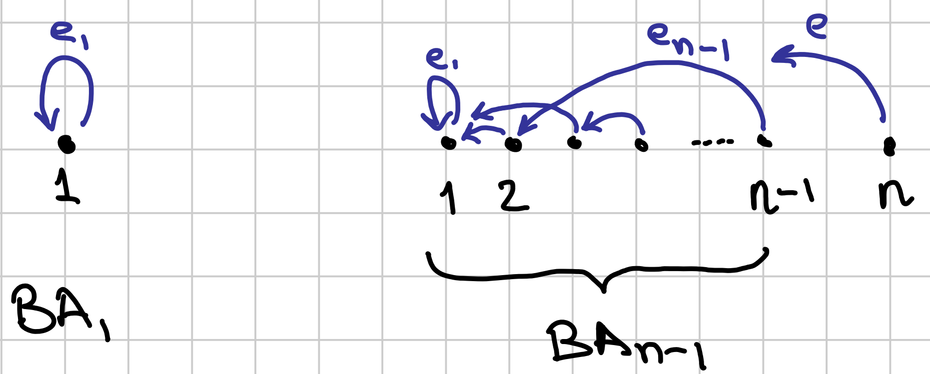
to appear ICALP 2017

MAX-PLANCK-GESELLSCHAFT

max planck institut informatik

Barabási-Albert Preferential Attachment Model [99]

- random graph model
- obtained by a random process



$$\text{head}(e_n) \propto \{\deg(i)\}_{i=1}^{n-1}$$

$$\Pr[h(e_n) = i] = \frac{1}{2(n-1)} \cdot \deg(i)$$

Construction in $O(n)$ [BB05, KRRSTU00, NLKB11]

Parallel [AKM13], Ext. Mem. [MP16]

- [BB05] Vladimir Batagelj and Ulrik Brandes. Efficient generation of large random networks. Physical Review E, 71(3):036113, 2005
- [NLKB11] Sadegh Nobari, Xuesong Lu, Panagiotis Karras, and Stéphane Bressan. Fast random graph generation. In Proceedings of the 14th international conference on extending database technology, pages 331–342. ACM, 2011
- [MP16] Ulrich Meyer and Manuel Penschuck. Generating massive scale-free networks under resource constraints. In Proceedings of the Eighteenth Workshop on Algorithm Engineering and Experiments, ALENEX 2016

SIAM J. Sci. Comput., 36(5), C424–C452. (29 pages) 2014

Software and High-Performance Computing

A Scalable Generative Graph Model with Community Structure

Tamara G. Kolda, **Ali Pinar**, **Todd Plantenga**, and **C. Seshadhri**

DOI:10.1137/130914218

The majority of graph models add edges one at a time in a way that each random edge influences the formation of future edges, making them inherently serial and therefore unscalable. The classic example is Preferential Attachment [2], but there are a variety of related models, e.g., [25, 28]. These models are more focused on

"Prediction is very difficult, especially about the future."

[Niels Bohr, Yogi Berra]

what is our answer to this in the BA context? ...

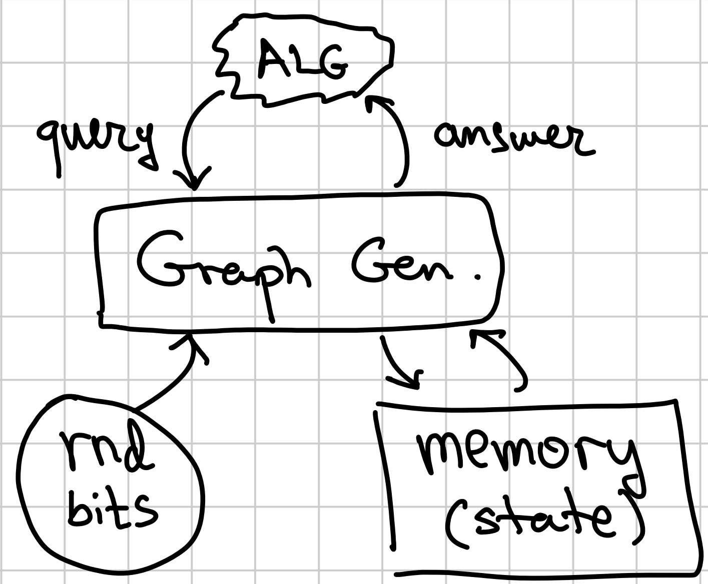
Graph Generator (for adjacency list queries)

adj. list sorted in asc. order

8 : 5, 13, 21, 1007

query : next neigh. of v

consistent : $x \in \text{list}(y) \Leftrightarrow y \in \text{list}(x)$



if answered 5, 13, 21 to 3 Q's for vertex 8

then [1,4], [6,12], [14,20] NOT neigh.

but [22, n] might be.

Main Result

Las Vegas BA_n graph generator.

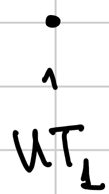
with probability $1 - \frac{1}{poly(n)}$ each query:

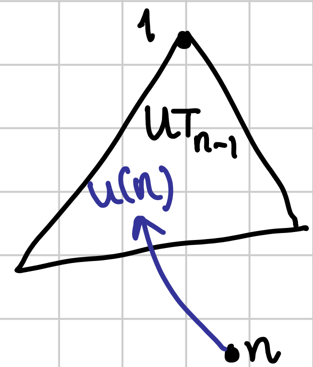
- running time $O(\log^5 n)$
- random bits $O(\log^4 n)$
- increase space $O(\log^3 n)$ bits.

local comp, not serial, (almost) scalable !

Recursive Trees

- Random graph model over in-trees rooted at 1.



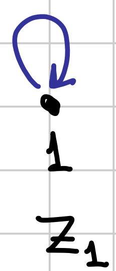


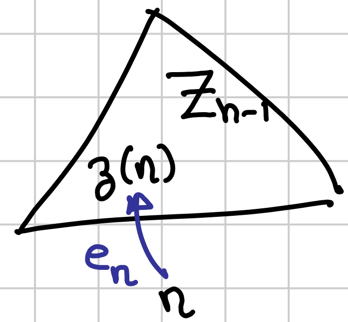
$$u(n) \propto \text{uniform}([n-1])$$

$$\Pr(u(n)=i) = \frac{1}{n-1}$$

Evolving Copying Model [KRRSTU00]

$d = 1$ deg
$\alpha = 1/2$ copyfac





$$z(n) = \begin{cases} u(n) & \text{if } b(n) = 1 \\ z(u(n)) & \text{if } b(n) = 0 \end{cases}$$

$$u(n) \propto \text{unif}([n-1])$$

$$b(n) \propto \text{unif}(\{0,1\})$$

Claim: Z_n & BA_n distributed identically

Idea: $\Pr(n \xrightarrow{BA} i) = \frac{1}{2(n-1)} \cdot \deg(i, BA_{n-1})$

$$\Pr(n \xrightarrow{Z_n} i) = \frac{1}{2} \cdot \frac{1}{n-1} + \frac{1}{2} \frac{\deg_{IN}(i, Z_{n-1})}{n-1}$$

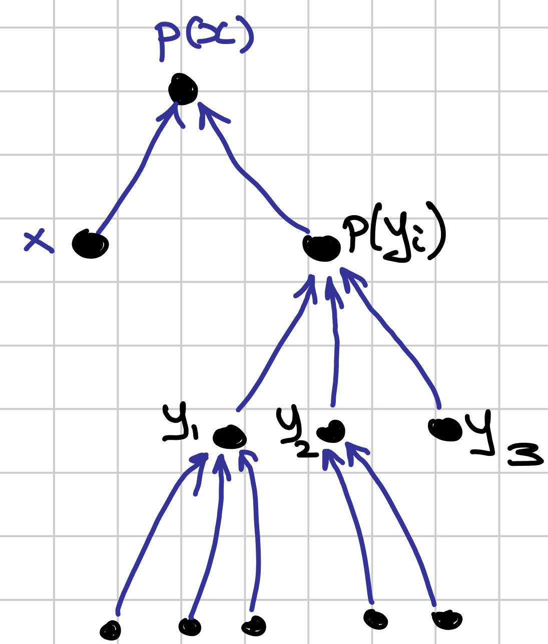
[KRRSTU00] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Andrew Tomkins, and Eli Upfal. Random graph models for the web graph. In 41st Annual Symposium on Foundations of Computer Science, FOCS 2000

Graph Gen. for rooted in-trees

Queries:

- parent of x.
- next-child of x

  (asc. order of children)



Suffices for BA_n

($p(x)$, children of x)

parent queries [KRRSTU00, AKM13]

$$p(i) = \begin{cases} u(i) & \text{if } b(i) = 1 \\ p(u(i)) & \text{if } b(i) = 0 \end{cases}$$

lazy filling of array $\{p(i)\}_{i=1}^{n}$

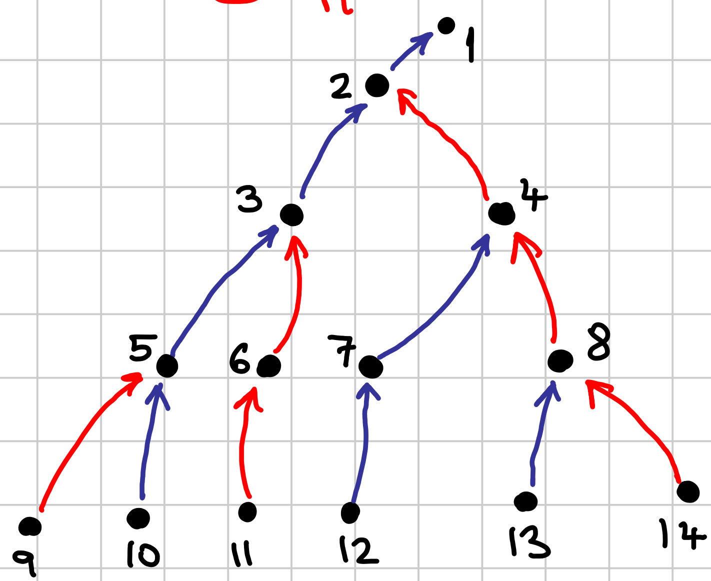
Claim: w.p $1 - \frac{1}{\text{poly}(n)}$ recursion depth $O(\log n)$

time $= O(\log^2 n)$, rnd $= O(\log^2 n)$, space $= O(\log^2 n)$

[KRRSTU00] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Andrew Tomkins, and Eli Upfal. Random graph models for the web graph. In 41st Annual Symposium on Foundations of Computer Science, FOCS 2000

[AKM13] Md. Maksudul Alam, Maleq Khan, and Madhav V. Marathe. Distributed-memory parallel algorithms for generating massive scale-free networks using preferential attachment model. In International Conference for High Performance Computing, Networking, Storage and Analysis, 2013

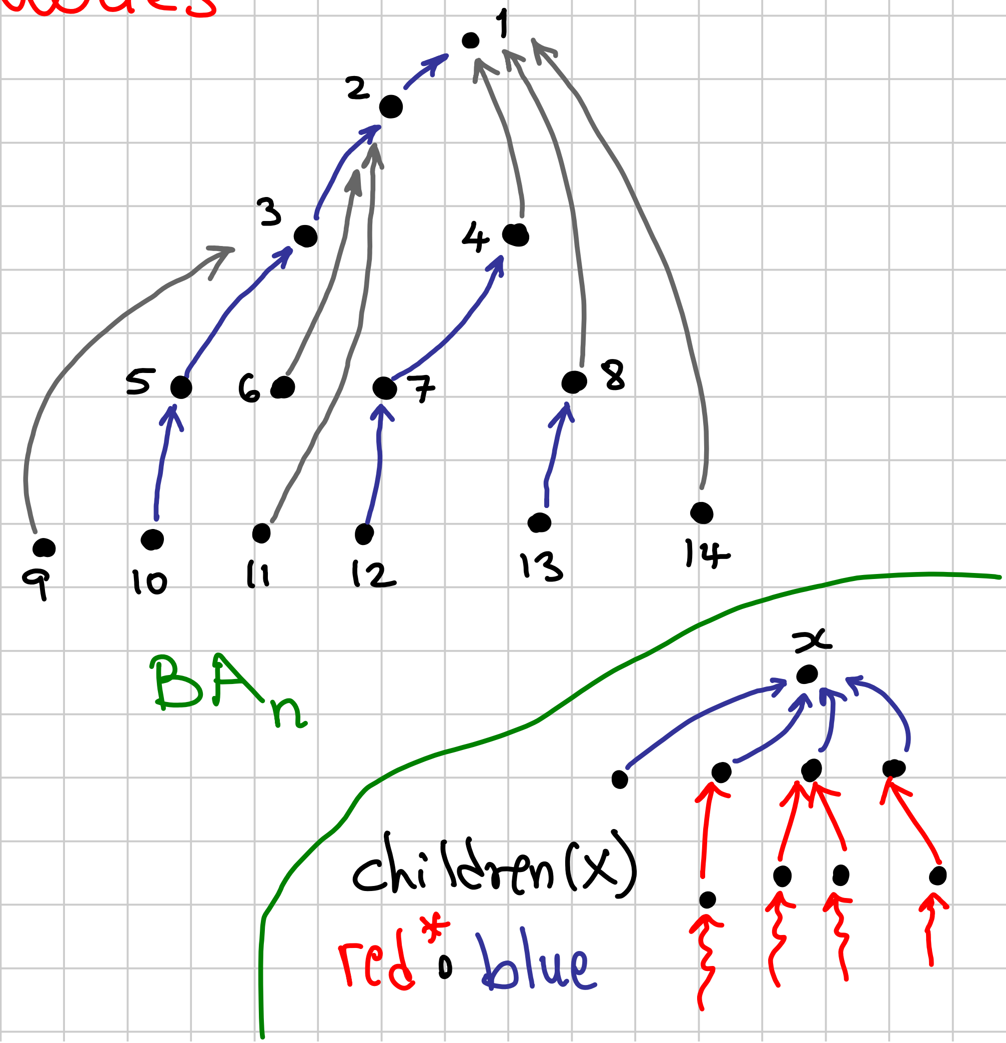
BA_n: next-child queries



Recursive Tree

blue $i \to j$ $b(i) = 1$

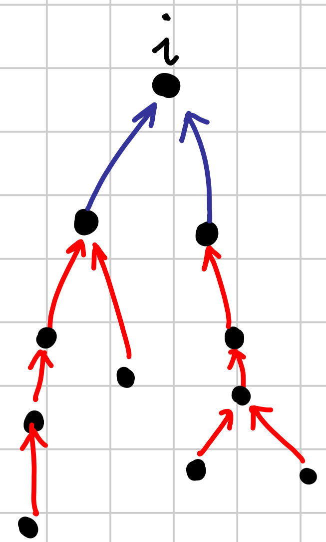
red $i \to j$ $b(i) = 0$



BA_n

red* ∘ blue

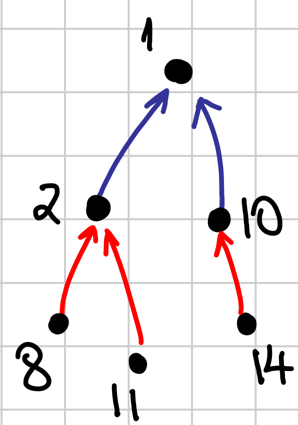
BA-children(i)

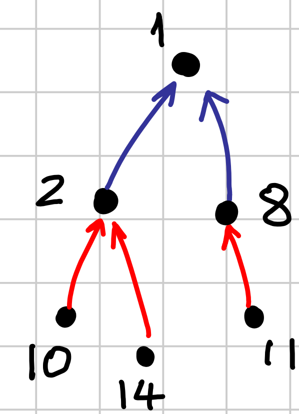


T_i red*∘blue tree hanging from i

Goal: output T_i in asc. order.

* T_i is a sub-tree of colored rec. tree.
* T_i is a heap ($p(x) < x$) & ord. siblings.

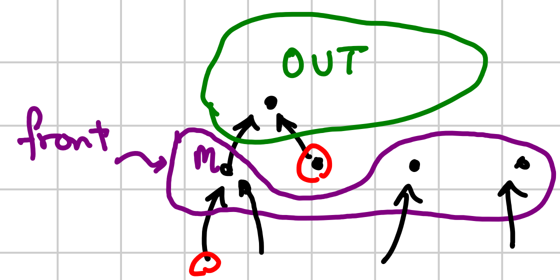




Scanning a Heap T_i (with ord. siblings)

Oracles: $p(x)$ & next-child(x) in T_i

derive: first-child(x), next-sibling$(x) \triangleq$ next-child$(p(x))$



$\text{front} \leftarrow \{i\}$

$m \leftarrow \min\{x : x \in \text{front}\}$

$\text{front} \leftarrow \text{front} \cup \{\text{first-child}(m), \text{next-sibling}(m)\} \setminus \{m\}$

return m

claim: $|\text{front}| \leq |\text{output}| + 1$

next-child in T_i

To implement $y = \text{next-child}(x, T_i)$

suffices to implement

$y = \text{next-child}(x, \text{Rec. Tree})$

& check that color of (y, x) is "good"

$\Rightarrow$

focus on oracle: $\text{next-child}(x, \text{rec. tree})$

next-child(x, rec. tree)

$$\Pr[u(y)=x] = \frac{1}{y-1}$$

$x \longleftarrow y$

Naïve: y = last child $(x)+1$

while $y \leq n$ do

- if $u(y) = \text{nil}$ then
  - flip bit $c_x(y)$ w.p. $\frac{1}{y-1}$
  - if $c_x(y) = 1$:
    - $u(y) = x$, return(y)
- else $y \leftarrow y+1$

next-child(x, rec. tree)

$\Pr[u(y)=x] = \frac{1}{[y-1]}$

$x \xleftarrow{} y$

Naive: y = last child $(x)+1$

while $y \leq n$ do

if $u(y) = nil$ then

flip bit $c_x(y)$ w.p. $\frac{1}{[y-1]}$

if $c_x(y) = 1$:

$u(y) = x$, return(y)

else $y \leftarrow y+1$

if $u(y) \neq nil$

$u(y) = x$: return(y)

$u(y) \neq x$: skip y.

next-child (x, rec. tree)

Naive: $y = \text{last child}(x) + 1$

while $y \leq n$ do

if $u(y) = \text{nil}$ then

flip bit $c_x(y)$ w.p. $\frac{1}{[y-1]}$

wrong! $u(y) \neq z$

if $c_x(y) = 1$:

$u(y) = x$, return(y)

else $y \leftarrow y + 1$

if $u(y) \neq \text{nil}$

$u(y) = x$: return(y)

$u(y) \neq x$: skip y.

$\Pr[u(y) = x] \neq \frac{1}{[y-1]}$

Obstacle 1

assume only next-child (1, Rec. Tree) oracles.

and last child (1) $= n/2$.

Coin prob. $\frac{1}{n/2}, \frac{1}{n/2+1}, \ldots$

$\Rightarrow \Omega(n)$ coin flips before stopping.

req. "nice" marginals

Solution: roll dice with $\frac{n}{2}+1$ sides

$$\mathrm{Prob}(\text{side } i) = \begin{cases} \frac{\frac{n}{2}-1}{(i-1)(i-2)} & \text{if } \frac{n}{2} < i \leq n \\ \frac{n/2-1}{n-1} & \text{if } i = n+1 \end{cases}$$

$\begin{cases} \text{rnd bits} \leq \log n \\ \text{time } O(\log^2 n) \end{cases}$

Obstacle 2

$\mathrm{Prob}(u(y) = x) \neq \frac{1}{y-1}$

marginal $= \frac{1}{|\Phi(y)|}$

Potential parents of y: $\Phi(y) \triangleq \{ i \mid \mathrm{last}(i) < y \}$

1) How to compute $|\Phi(y)|$?

req. "nice" marginals

2) Roll dice to find next child.
(instead of seq. tossing of coins)

Solution

- Impose assumptions so that

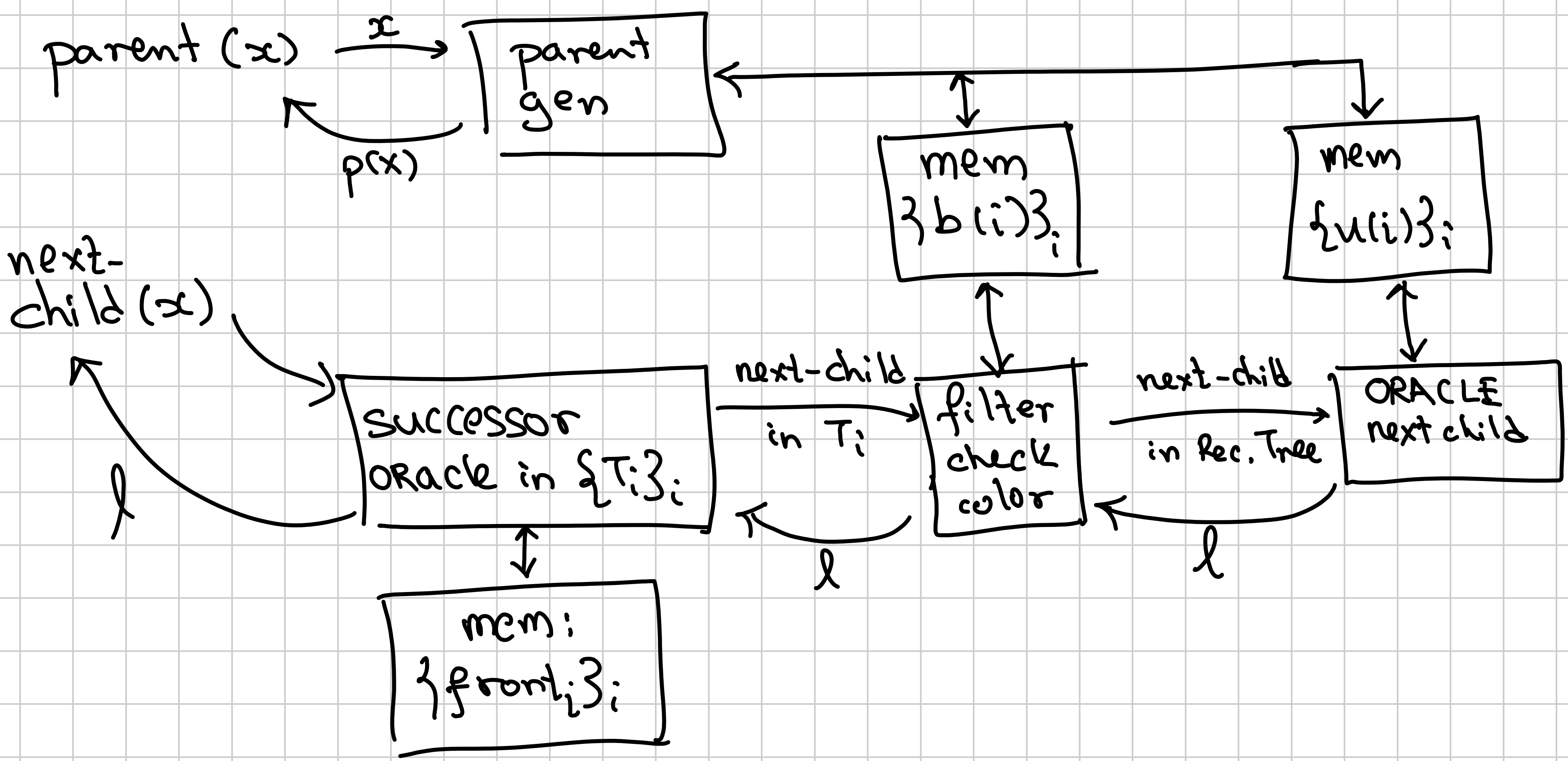
$$\Phi(y+1) - \Phi(y) \leq 1$$

- Manage set K so that

$$\phi(y) = \phi(y+1) \iff y \in K$$

- Obtain harmonic behavior $\frac{1}{w}, \frac{1}{w+1}, \dots$ of marginals (coins are emulated by dice)

BA_n Generator Overview



OPEN

1) Graph generators for other models?

2) Generators for other random processes?

3) Example of lower bound:
find random graph model with no sublinear generator.

Higher Out-Degrees

Bollobas & Riordan [04]

$BA_{n,deg(d)} \equiv$ Coalescing $[i, i+(d-1)]$ in BA_{nd} into a single node $\hat{i}$


That's all Folks!
SUPERB wallpapers