A Generalization of Spira's Theorem and Circuits with Small Segregators or Separators

Anna Gál, Jing-Tang Keith Jang

Department of Computer Science
University of Texas at Austin

January 25, 2012

Definitions

- Boolean circuit: a DAG with nodes labeled by $\{\vee, \wedge, \neg\}$
  Assume fan-in ≤ 2

Definitions

- Boolean circuit: a DAG with nodes labeled by $\{\vee, \wedge, \neg\}$
  Assume fan-in ≤ 2
- Boolean formula: tree-like circuit, i.e. fan-out ≤ 1

Definitions

- Boolean circuit: a DAG with nodes labeled by $\{\vee, \wedge, \neg\}$
  Assume fan-in ≤ 2
- Boolean formula: tree-like circuit, i.e. fan-out ≤ 1
- Size: the number of gates

Definitions

- Boolean circuit: a DAG with nodes labeled by $\{\vee, \wedge, \neg\}$
  Assume fan-in ≤ 2
- Boolean formula: tree-like circuit, i.e. fan-out ≤ 1
- Size: the number of gates
- Depth: length of the longest path from inputs to output

Spira's Theorem

Theorem

Spira [1971]

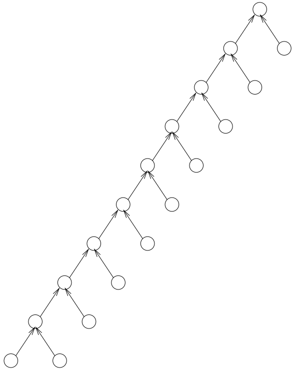
Let F be any Boolean formula of size s. Then F can be simulated by an equivalent formula of depth $O(\log s)$.

Spira's Theorem

Theorem

Spira [1971]

Let F be any Boolean formula of size s. Then F can be simulated by an equivalent formula of depth $O(\log s)$.

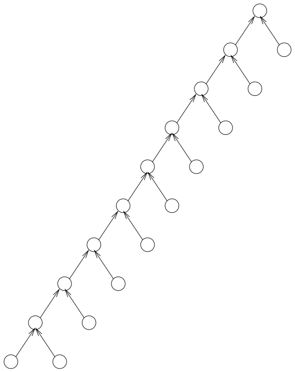


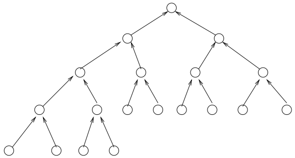
Spira's Theorem

Theorem

Spira [1971]

Let F be any Boolean formula of size s. Then F can be simulated by an equivalent formula of depth $O(\log s)$.





Spira's Theorem for General Circuits?

- Wide open for general circuits

Spira's Theorem for General Circuits?

- Wide open for general circuits
- Current best result: $O(s/\log s)$ depth simulation by Patterson and Valiant [1976], Dymond and Tompa [1985]

Spira's Theorem for General Circuits?

- Wide open for general circuits
- Current best result: $O(s/\log s)$ depth simulation by Patterson and Valiant [1976], Dymond and Tompa [1985]
- Time vs. space: Better general simulation of size by depth in the uniform setting will improve the simulation of (deterministic) *TIME* by *SPACE*.

Spira's Theorem for General Circuits?

- Wide open for general circuits
- Current best result: $O(s/\log s)$ depth simulation by Patterson and Valiant [1976], Dymond and Tompa [1985]
- Time vs. space: Better general simulation of size by depth in the uniform setting will improve the simulation of (deterministic) *TIME* by *SPACE*.
- Sequential time vs. parallel time: Better general simulation of size by depth will improve the simulation of *TIME* by logcost PRAM.

Separator and Segregator

Informally,

- Separator: a subset of nodes whose removal yields two subgraphs of comparable sizes

Separator and Segregator

Informally,

- Separator: a subset of nodes whose removal yields two subgraphs of comparable sizes
- Segregator: a relaxed version of separators for DAGs

Separator and Segregator

Informally,

- Separator: a subset of nodes whose removal yields two subgraphs of comparable sizes
- Segregator: a relaxed version of separators for DAGs
- Paul, Pippenger, Szemerédi, and Trotter [1983] and Santhanam [2000] used segregators to study the computation graphs of Turing machines

Definition: Separators

Definition

- *A* separator of size k *of a DAG* $G = (V, E)$ *is a set of* k *nodes* $S \subseteq V$ *such that the removal of* S *partitions* $G \setminus S$ *into two subDAGs,* G_1 *and* G_2*, such that* $|G_i| \leq \frac{2}{3}|V|$ *for* $i = 1, 2$*, and there are no edges either from* G_1 *to* G_2*, or from* G_2 *to* G_1 *in* $G \setminus S$*.*

Definition: Separators

Definition

- *A separator of size k of a DAG $G = (V, E)$ is a set of k nodes $S \subseteq V$ such that the removal of S partitions $G \setminus S$ into two subDAGs, G_1 and G_2, such that $|G_i| \leq \frac{2}{3}|V|$ for $i = 1, 2$, and there are no edges either from G_1 to G_2, or from G_2 to G_1 in $G \setminus S$.*
- *A Boolean circuit C has separators of size $f()$ if the underlying DAG of every subcircuit of C with s gates has a separator of size at most $f(s)$.*

Definition: Segregators

Definition

- Paul, Pippenger, Szemerédi and Trotter [1983]
  A *segregator of size k* of a DAG $G = (V, E)$ is a set of k nodes $S \subseteq V$ such that every node in $G \setminus S$ has at most $\frac{2}{3}|V|$ predecessors in $G \setminus S$.

Definition: Segregators

Definition

- Paul, Pippenger, Szemerédi and Trotter [1983]
  A *segregator of size k* of a DAG $G = (V, E)$ is a set of k nodes $S \subseteq V$ such that every node in $G \setminus S$ has at most $\frac{2}{3}|V|$ predecessors in $G \setminus S$.
- A Boolean circuit C has segregators of size $f()$ if the underlying DAG of every subcircuit of C with s gates has a segregator of size at most $f(s)$.

Definition: Segregators

Definition

- Paul, Pippenger, Szemerédi and Trotter [1983]
  A *segregator of size k* of a DAG $G = (V, E)$ is a set of k nodes $S \subseteq V$ such that every node in $G \setminus S$ has at most $\frac{2}{3}|V|$ predecessors in $G \setminus S$.
- A Boolean circuit C has segregators of size $f()$ if the underlying DAG of every subcircuit of C with s gates has a segregator of size at most $f(s)$.

Lemma

Let G be a DAG. If G has a separator of size k, then G also has a segregator of size at most k.

Definition: Segregators

Definition

- Paul, Pippenger, Szemerédi and Trotter [1983]
  A *segregator of size k* of a DAG $G = (V, E)$ is a set of k nodes $S \subseteq V$ such that every node in $G \setminus S$ has at most $\frac{2}{3}|V|$ predecessors in $G \setminus S$.
- A Boolean circuit C has segregators of size $f()$ if the underlying DAG of every subcircuit of C with s gates has a segregator of size at most $f(s)$.

Lemma

Let G be a DAG. If G has a separator of size k, then G also has a segregator of size at most k.

From now on we focus on segregators.

Spira's Theorem: History

- Wegener [1983] extended Spira's theorem to monotone Boolean formulas

Spira's Theorem: History

- Wegener [1983] extended Spira's theorem to monotone Boolean formulas
- Bonet and Buss [1994] improved the constants in the depth bounds and the size of the simulation for Boolean formulas

Spira's Theorem: History

- Wegener [1983] extended Spira's theorem to monotone Boolean formulas
- Bonet and Buss [1994] improved the constants in the depth bounds and the size of the simulation for Boolean formulas
- Brent [1974], Bshouty, Cleve, Eberly [1995] extended Spira's theorem to arithmetic formulas

Spira's Theorem: History

- Wegener [1983] extended Spira's theorem to monotone Boolean formulas
- Bonet and Buss [1994] improved the constants in the depth bounds and the size of the simulation for Boolean formulas
- Brent [1974], Bshouty, Cleve, Eberly [1995] extended Spira's theorem to arithmetic formulas
- All these results study formulas, i.e. tree-like circuits with fan-out 1

Recent Advances

Recall:

Theorem

(Re-Phrased) Spira [1971] The class of languages decided by non-uniform families of polynomial-size tree-like circuits equals non-uniform NC^1.

Recent Advances

Recall:

Theorem

(Re-Phrased) Spira [1971] The class of languages decided by non-uniform families of polynomial-size tree-like circuits equals non-uniform NC^1.

Theorem

Jansen and Sarma [2010]
Boolean circuits with size $n^{O(1)}$ and constant treewidth k can be simulated by circuits of depth $O(k^2 \log n)$, if k and tree-decomposition are given along with circuit description.

Recent Advances

Recall:

Theorem

(Re-Phrased) Spira [1971] The class of languages decided by non-uniform families of polynomial-size tree-like circuits equals non-uniform NC^1.

Theorem

Jansen and Sarma [2010]
Boolean circuits with size $n^{O(1)}$ and constant treewidth k can be simulated by circuits of depth $O(k^2 \log n)$, if k and tree-decomposition are given along with circuit description.

Corollary

Jansen and Sarma [2010]
The class of languages with non-uniform polynomial-size and bounded treewidth circuit families equals non-uniform NC^1.

Our Generalization of Spira's Theorem

Our improvement:

Theorem

The class of languages decided by non-uniform families of polynomial-size circuits with constant-size segregators equals non-uniform NC^1.

Our Generalization of Spira's Theorem

Our improvement:

Theorem

The class of languages decided by non-uniform families of polynomial-size circuits with constant-size segregators equals non-uniform NC^1.

More generally,

Theorem

Generalization of Spira's Theorem

Any Boolean circuit of size s with segregators of size $f()$ can be simulated in depth $O(f(s)\log s)$. Moreover, if $f(s) = \Omega(s^\epsilon)$ for some constant ϵ, then the simulating circuit has depth $O(f(s))$.

Definition: Uniform Circuit Families

Definition

A family of Boolean circuits $\{C_n\}$ is called h(n)-space uniform, if there exists a deterministic Turing machine M that on input 1^n, outputs the standard description of C_n using space $O(h(n))$ for all n. In particular, $\{C_n\}$ is logspace uniform if $h(n) = \log n$.

Our Result: Uniform Simulation

Theorem

Uniform Generalization of Spira's Theorem

Let

- $\mathcal{C}$: $h(n)$-space uniform family of Boolean circuits
- $C_n \in \mathcal{C}$: has n inputs, size $s = s(n)$, and segregators of size $f()$
- $g(s) = f(s)$ if $f(s) = \Omega(s^c)$ for some constant $c > 0$, and $f(s) \log s$ otherwise

Then $\mathcal{C}$ *can be simulated by a* $O(h(n) + g(s) \log s)$*-space uniform family of Boolean circuits of depth* $O(g(s))$.

Our Result: Uniform Simulation

Theorem

Uniform Generalization of Spira's Theorem

Let

- $\mathcal{C}$: *$h(n)$-space uniform family of Boolean circuits*
- $C_n \in \mathcal{C}$: *has n inputs, size $s = s(n)$, and segregators of size $f()$*
- $g(s) = f(s)$ *if* $f(s) = \Omega(s^c)$ *for some constant* $c > 0$, *and* $f(s) \log s$ *otherwise*

Then $\mathcal{C}$ can be simulated by a $O(h(n) + g(s) \log s)$-space uniform family of Boolean circuits of depth $O(g(s))$.

- Our generalization of Spira's theorem is $O(h(n) + g(s) \log s)$-space uniform.

Our Result: Uniform Simulation

Theorem

Uniform Generalization of Spira's Theorem

Let

- $\mathcal{C}$: *h(n)-space uniform family of Boolean circuits*
- $C_n \in \mathcal{C}$: *has n inputs, size* $s = s(n)$*, and segregators of size* $f()$
- $g(s) = f(s)$ *if* $f(s) = \Omega(s^c)$ *for some constant* $c > 0$*, and* $f(s) \log s$ *otherwise*

Then $\mathcal{C}$ *can be simulated by a* $O(h(n) + g(s) \log s)$*-space uniform family of Boolean circuits of depth* $O(g(s))$*.*

- Our generalization of Spira's theorem is $O(h(n) + g(s) \log s)$-space uniform.
- It is still unknown if Spira's re-structuring algorithm can be done in logspace.

Circuit Value Problem

- Uniform simulation is relevant to the Circuit Value Problem.

Circuit Value Problem

- Uniform simulation is relevant to the Circuit Value Problem.
- Circuit Value Problem: Given the description of a Boolean circuit C and an input x to C, what is the value of $C(x)$?

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]
- Monotone Planar Circuit Value

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]
- Monotone Planar Circuit Value
  - $O(\log^3 n)$ space ($O(\log^2 n)$ EREW time by Ramachandran and Yang [1996])

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in *SPACE*($\log n$) by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]
- Monotone Planar Circuit Value
  - $O(\log^3 n)$ space ($O(\log^2 n)$ EREW time by Ramachandran and Yang [1996])
  - *LOGDCFL* by Barrington, Lu, Miltersen, and Skyum [1999]

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]
- Monotone Planar Circuit Value
  - $O(\log^3 n)$ space ($O(\log^2 n)$ EREW time by Ramachandran and Yang [1996])
  - $LOGDCFL$ by Barrington, Lu, Miltersen, and Skyum [1999]
- Comparator Circuit Value: $O(\sqrt{n} poly \log(n))$ space ($O(\sqrt{n} poly \log(n))$ PRAM time by Mayr and Subramanian [1992])

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]
- Monotone Planar Circuit Value
  - $O(\log^3 n)$ space ($O(\log^2 n)$ EREW time by Ramachandran and Yang [1996])
  - *LOGDCFL* by Barrington, Lu, Miltersen, and Skyum [1999]
- Comparator Circuit Value: $O(\sqrt{n} poly \log(n))$ space ($O(\sqrt{n} poly \log(n))$ PRAM time by Mayr and Subramanian [1992])
- Square Circuit Value: $O(\sqrt{n} poly \log(n))$ space ($O(\sqrt{n} poly \log(n))$ PRAM time by Condon [1994])

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]
- Monotone Planar Circuit Value
  - $O(\log^3 n)$ space ($O(\log^2 n)$ EREW time by Ramachandran and Yang [1996])
  - $LOGDCFL$ by Barrington, Lu, Miltersen, and Skyum [1999]
- Comparator Circuit Value: $O(\sqrt{n} poly \log(n))$ space ($O(\sqrt{n} poly \log(n))$ PRAM time by Mayr and Subramanian [1992])
- Square Circuit Value: $O(\sqrt{n} poly \log(n))$ space ($O(\sqrt{n} poly \log(n))$ PRAM time by Condon [1994])

All these CVP can be solved in sublinear space.

History: Circuit Value Problem

- Boolean Formula Value (input formula is in sentence form)
  - in $SPACE(\log n)$ by Lynch [1977]
  - in DLOGTIME-uniform NC_1 by Buss, Cook, Gupta, and Ramachandran [1992]
- Monotone Planar Circuit Value
  - $O(\log^3 n)$ space ($O(\log^2 n)$ EREW time by Ramachandran and Yang [1996])
  - $LOGDCFL$ by Barrington, Lu, Miltersen, and Skyum [1999]
- Comparator Circuit Value: $O(\sqrt{n} poly \log(n))$ space ($O(\sqrt{n} poly \log(n))$ PRAM time by Mayr and Subramanian [1992])
- Square Circuit Value: $O(\sqrt{n} poly \log(n))$ space ($O(\sqrt{n} poly \log(n))$ PRAM time by Condon [1994])

All these CVP can be solved in sublinear space.
But none of these problems is believed to be P-complete.

Our Result: Planar Circuit Value Problem

Theorem

The Planar Circuit Value Problem can be decided in $SPACE(\sqrt{n}\log n)$.

Our Result: Planar Circuit Value Problem

Theorem

The Planar Circuit Value Problem can be decided in SPACE$(\sqrt{n}\log n)$.

- Goldschlager [1977] Planar Circuit Value is P-complete.

Our Result: Planar Circuit Value Problem

Theorem

The Planar Circuit Value Problem can be decided in $SPACE(\sqrt{n}\log n)$.

- Goldschlager [1977] Planar Circuit Value is P-complete.
- No sublinear space nor sublinear parallel time algorithm is currently known for CVP over general circuits.

Proofs

Spira's Construction

Lemma

Jordan [1869]

A tree T has a node w whose removal divides T into two subtrees T_1 and T_2, such that the sizes of both subtrees are at most $2/3$ of the size of T.

Spira's Construction

Lemma

Jordan [1869]

A tree T has a node w whose removal divides T into two subtrees T_1 and T_2, such that the sizes of both subtrees are at most $2/3$ of the size of T.

Spira's re-structuring formula:

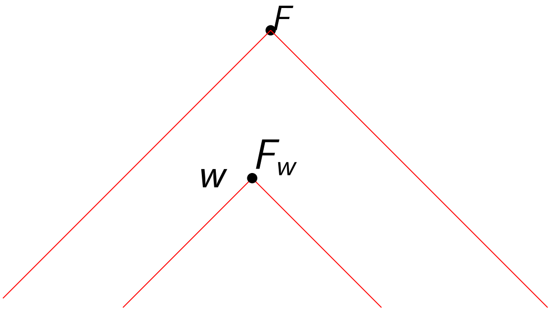
$$F(x) = (F_1(x) \wedge F_w(x)) \vee (F_0(x) \wedge \neg F_w(x)),$$

where w is the node in Jordan's lemma, and

- F_0: function computed by the original formula with w replaced by 0
- F_1: function computed by the original formula with w replaced by 1
- F_w: function computed at w

Spira's Construction

Original circuit:



F: function computed by the circuit

w: separator for the tree

F_w: function computed at w

Spira's Construction

$$F(x) = (F_1(x) \wedge F_w(x)) \vee (F_0(x) \wedge \neg F_w(x)),$$

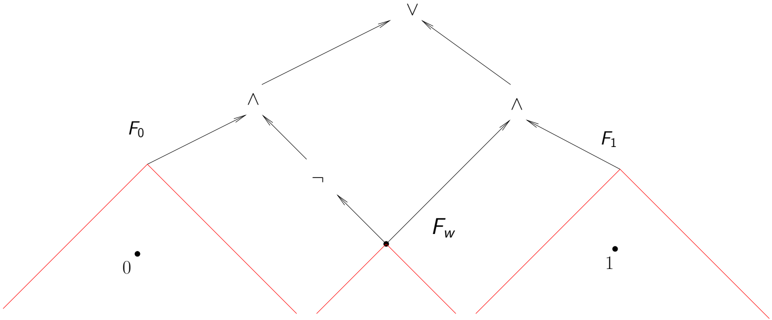
where w is the node in Jordan's lemma.

Spira's Construction

$$F(x) = (F_1(x) \wedge F_w(x)) \vee (F_0(x) \wedge \neg F_w(x)),$$

where w is the node in Jordan's lemma.

New circuit:



F_0: function computed by the original formula with w replaced by 0

F_1: function computed by the original formula with w replaced by 1

F_w: function computed at w

Proof of Our Generalization of Spira's Theorem

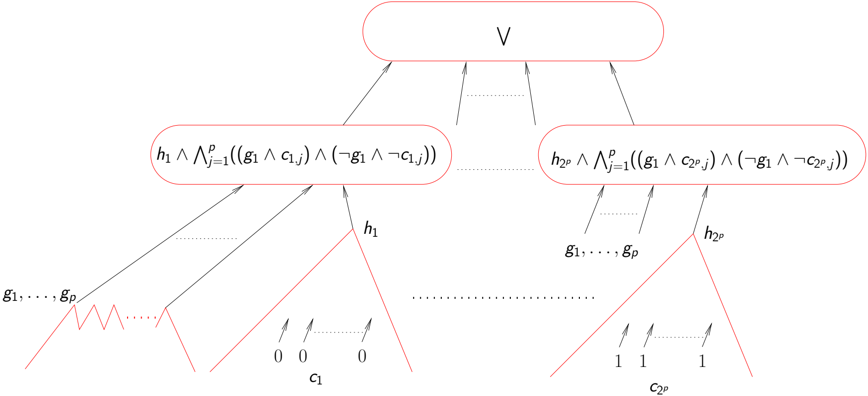
Generalization for circuits with segregators of size $p = f(s)$:

$$\bigvee_{i=1}^{2^p} \left(h_i \wedge \bigwedge_{j=1}^{p} ((g_j \wedge c_{i,j}) \vee (\neg g_j \wedge \neg c_{i,j})) \right),$$

where

- g_j is the function computed at v_j, the jth node in the segregator
- $c_{i,j}$ is the jth bit of the ith Boolean vector of length p $(i = 1 \ldots 2^p)$
- h_i is the function computed by the original circuit after v_j is replaced by $c_{i,j}$ for every $j = 1 \ldots p$

Generalized Construction



p: size of the segregator

c_i: the ith Boolean vector of length p

$c_{i,j}$: the jth component in the ith Boolean vector of length p

h_i: function computed by the original circuit with $v_1, \ldots, v_p$ replaced by c_i

g_i: function computed at v_i

$v_1, \ldots, v_p$: segregator for the circuit

Proof of Uniform Generalization of Spira's Theorem

Recall:

Theorem

Uniform Generalization of Spira's Theorem

Let

- $\mathcal{C}$*: $h(n)$-space uniform family of Boolean circuits*
- $C_n \in \mathcal{C}$*: has n inputs, size $s = s(n)$, and segregators of size $f()$*
- $g(s) = f(s)$ *if* $f(s) = \Omega(s^c)$ *for some constant* $c > 0$*, and* $f(s) \log s$ *otherwise*

Then $\mathcal{C}$ can be simulated by a $O(h(n) + g(s) \log s)$-space uniform family of Boolean circuits of depth $O(g(s))$.

Outline of Proof

The uniform simulation consists of the following recursive steps.

1. Compute a minimum segregator in small space

Outline of Proof

The uniform simulation consists of the following recursive steps.

1. Compute a minimum segregator in small space
2. Generate the simulating circuits based on the computed segregator in small space (using our generalization of Spira's Theorem)

Outline of Proof

The uniform simulation consists of the following recursive steps.

1. Compute a minimum segregator in small space
2. Generate the simulating circuits based on the computed segregator in small space (using our generalization of Spira's Theorem)

Note: The only input is the description of the circuit to be simulated. Size of the minimum segregator is not known in advance.

Generating the Simulating Circuits in Small Space

In each recursive step, perform:

Generating the Simulating Circuits in Small Space

In each recursive step, perform:

- Replace gates in the current segregator by constants

Generating the Simulating Circuits in Small Space

In each recursive step, perform:

- Replace gates in the current segregator by constants
- Generate the current simulating subcircuit using our generalization of Spira's Theorem

Generating the Simulating Circuits in Small Space

In each recursive step, perform:

- Replace gates in the current segregator by constants
- Generate the current simulating subcircuit using our generalization of Spira's Theorem

During the recursion, the following information must be retained on tape:

Generating the Simulating Circuits in Small Space

In each recursive step, perform:

- Replace gates in the current segregator by constants
- Generate the current simulating subcircuit using our generalization of Spira's Theorem

During the recursion, the following information must be retained on tape:

- Relevant information about the segregators in the previous steps

Generating the Simulating Circuits in Small Space

In each recursive step, perform:

- Replace gates in the current segregator by constants
- Generate the current simulating subcircuit using our generalization of Spira's Theorem

During the recursion, the following information must be retained on tape:

- Relevant information about the segregators in the previous steps
- List of values substituted for nodes in the segregators in the previous steps

Generating the Simulating Circuits in Small Space

In each recursive step, perform:

- Replace gates in the current segregator by constants
- Generate the current simulating subcircuit using our generalization of Spira's Theorem

During the recursion, the following information must be retained on tape:

- Relevant information about the segregators in the previous steps
- List of values substituted for nodes in the segregators in the previous steps

All can be done in $O(h(n) + f(s) \log s + \log^2 s)$ space

Circuit Value Problem for Circuits with Small Segregators

Theorem

The Circuit Value problem for circuits of size s with segregators of size $f()$ is in $SPACE(f(s)\log s)$ if $f(s) = \Omega(s^{\varepsilon})$ for some constant $\varepsilon > 0$, and $SPACE(f(s)\log^2 s)$ otherwise.

Circuit Value Problem for Circuits with Small Segregators

Theorem

The Circuit Value problem for circuits of size s with segregators of size $f()$ *is in* $SPACE(f(s)\log s)$ *if* $f(s) = \Omega(s^{\varepsilon})$ *for some constant* $\varepsilon > 0$, *and* $SPACE(f(s)\log^2 s)$ *otherwise.*

Follows from

- Our uniform generalization of Spira's Theorem
- Borodin [1977] Any $h(n)$-space uniform circuit family of depth $h(n) \geq \log n$ can be simulated by $O(h(n))$-space Turing machines.

Corollaries

Theorem

The Boolean Circuit Value problem for circuits with constant-size segregators is in $SPACE(\log^2 n)$.

Corollaries

Theorem

The Boolean Circuit Value problem for circuits with constant-size segregators is in $SPACE(\log^2 n)$.

Theorem

The Planar Circuit Value Problem can be decided in $SPACE(\sqrt{n} \log n)$.

Corollaries

Theorem

The Boolean Circuit Value problem for circuits with constant-size segregators is in $SPACE(\log^2 n)$.

Theorem

The Planar Circuit Value Problem can be decided in $SPACE(\sqrt{n}\log n)$.

Follows from

- Our theorem in the previous slide
- Lipton and Tarjan [1976] Any planar graph of size s has a separator of size $O(\sqrt{s})$.

Open Questions

- $O(\log s)$ space uniform version of Spira's theorem (Currently we have $O(\log^2 s)$ space uniform)

Open Questions

- $O(\log s)$ space uniform version of Spira's theorem
(Currently we have $O(\log^2 s)$ space uniform)

- Generate polynomial-size simulating circuits

Open Questions

- $O(\log s)$ space uniform version of Spira's theorem (Currently we have $O(\log^2 s)$ space uniform)

- Generate polynomial-size simulating circuits
- Get rid of the $\log s$ factor from the recursion

The End

Savitch's Theorem

Theorem

Savitch [1970]

Given a directed graph G on s nodes and two nodes $u, v \in G$, there exists a deterministic Turing machine that decides if there is a path from u to v in G using space $O(\log^2 s)$.

Find a Minimum Size Segregator in Small Space

For graphs

Lemma

Given: a DAG G with s nodes such that G has a segregator of size $f(s)$
Then: $\exists$ a TM M_1 such that on input G, M_1 outputs a segregator of size at most $f(s)$ using space $O(f(s)\log s + \log^2 s)$.

Find a Minimum Size Segregator in Small Space

For graphs

Lemma

Given: a DAG G with s nodes such that G has a segregator of size $f(s)$
Then: $\exists$ a TM M_1 such that on input G, M_1 outputs a segregator of size at most $f(s)$ using space $O(f(s)\log s + \log^2 s)$.

Proof Sketch

1. *Given a node v, use Savitch's algorithm to count the number of predecessors of v.*
2. *Enumerate subsets of G in the order of size then in lex. Test if the subset is a segregator by counting the number of predecessors.*

Find a Minimum Size Segregator in Small Space

For uniform circuit families

Lemma

Given:

- $\mathcal{C}$: $h(n)$-space uniform family of circuits
- $C_n \in \mathcal{C}$: size $s = s(n)$ with a segregator of size $\leq f(s)$

Then:

$\exists$ a TM $\hat{M}$ such that on input 1^n, $\hat{M}$ outputs a segregator of C_n of size at most $f(s)$ using space $O(h(n) + f(s) \log s + \log^2 s)$.

Noga Alon, Paul Seymour, Robin Thomas: A Separator Theorem for Graphs with an Excluded Minor and its Applications. Proceedings of the ACM Symposium on Theory of Computing, 293–299 (1990)

M. Bonet and S. R. Buss: Size-depth tradeoffs for Boolean formulae. *Information Processing Letters*, 49(3), pp. 151–155 (1994).

D. A. M. Barrington, C.-J. Lu, P. B. Miltersen, and S. Skyum.
On monotone planar circuits.
In *IEEE Conference on Computational Complexity*, 24–33 (1999).

Allan Borodin: On Relating Time and Space to Size and Depth. SIAM Journal on Computing V.6 NO.4, 733–744 (1977)

R. P. Brent: The Parallel Evaluation of General Arithmetic Expressions. *Journal of the ACM*, 21(2), pp. 201–206 (1974).

N. Bshouty, R. Cleve and W. Eberly: Size-Depth Tradeoffs for Algebraic Formulas. *SIAM Journal on Computing*, 24(4), pp. 682–705 (1995).

S. Buss, S. Cook, A. Gupta, and V. Ramachandran.

An optimal parallel algorithm for formula evaluation.
SIAM Journal on Computing, 21(4):755–780, August 1992.

A. Condon: A Theory of Strict P-Completeness. Computational Complexity, V.4 (3) 220–241 (1994).

Patrick Dymond and Martin Tompa: Speedups of Deterministic Machines by Synchronous Parallel Machines. J. Comp. and Sys. Sci. V.30 NO.2, 149–161 (1985)

A. Gál and J. Jang: The size and depth of layered Boolean circuits. *Proceedings of LATIN 2010, LNCS 6034*, pp. 372–383 (2010).

Gaskov: The depth of Boolean functions. Probl. Kibernet. V.34, 265–268 (1978)

J.R. Gilbert, J.P. Hutchinson, R.E. Tarjan: A Separator Theorem for Graphs of Bounded Genus. Journal of Algorithms V.5 NO.3, 391–407 (1984)

L. Goldschlager.
The monotone and planar circuit value problem is complete for P.

SIGACT News, pages 25–27, 1977.

J. Hopcroft and W. Paul and L. Valiant: On Time Versus Space. Theory of Computation V.24 NO.2, 332–337 (1977)

M. Jansen and J. Sarma M. N.: Balancing Bounded Treewidth Circuits. *5th International Computer Science Symposium in Russia*, pp. 228–239 (2010).

C. Jordan.
Sur les assemblages des lignes.
Journal für die reine und angewandte Mathematik, 70:185–190, 1869.

R. Lipton and R.E. Tarjan: A Separator Theorem for Planar Graphs. SIAM J. Appl. Math. V.36, 177–189 (1979)

N. A. Lynch: Log space recognition and translation of parenthesis languages. *J. Assoc. Comput. Mach.*, 24, pp. 583–590 (1977).

Ernst W. Mayr and Ashok Subramanian. The Complexity of Circuit Value and Network Stability. Journal of Computer and System Sciences V.44 NO.2, 302–323 (1992)

W.F. McColl and M.S. Paterson: The depth of all Boolean functions. SIAM J. on Comp. V.6, 373–380 (1977)

M.S. Paterson and L.G. Valiant: Circuit Size is Nonlinear in Depth. Theoretical Computer Science V.2 NO.3, 397-400 (1976)

W. J. Paul, N. Pippenger, E. Szemerédi, and W. T. Trotter: On determinism versus non-determinism and related problems. *Proceedings of FOCS*, pp. 429–438 (1983).

W. Paul and R.E. Tarjan and J. Celoni: Space Bounds for a Game on Graphs. Mathematical Systems Theory V.10, 239–251 (1977)

N. Pippenger and M. Fischer: Relations Among Complexity Measures. Journal of the ACM V.26, 361–381 (1979)

Vijaya Ramachandran and Honghua Yang: An Efficient Parallel Algorithm for the General Planar Monotone Circuit Value Problem. SIAM Journal on Computing V.25 (2), 312–339 (1996).

- R. Santhanam: On separators, segregators and time versus space. *Proceedings of the Sixteenth Annual Conference on Computational Complexity*, pp. 286–294 (2000).
- W. J. Savitch: Relationships Between Nondeterministic and Deterministic Tape Complexities. *Journal of Computer and System Sciences*, 4(2), pp. 177–192 (1970).
- P.M. Spira: On time-hardware complexity tradeoffs for Boolean functions. In Proc. 4th Hawaii Symp. on System Sciences, 525–527 (1971)
- G. Turán: On restricted Boolean circuits. Fund. of Comp. Theory, 460–469 (1989)
- Jeffrey D. Ullman: Computational Aspects of VLSI. (1984)
- Jeffrey Scott Vitter and Roger A. Simons, New Classes for Parallel Complexity: A Study of Unification and Other Complete Problems for P. IEEE Trans. Computers, V.35 (5) 403–418 (1986)

Ingo Wegener: Relating monotone formula size and monotone depth of Boolean functions. *Information Processing Letters*, 16(1), pp. 41–42 (1983).