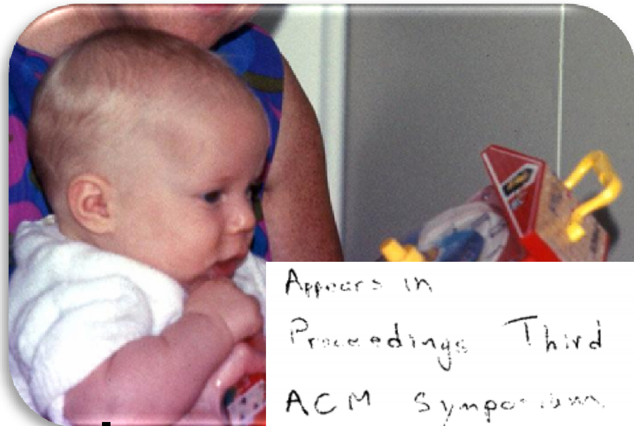
Research in Security and Privacy

David Evans
www.cs.virginia.edu/evans
www.jeffersonswheel.org

cs6190 - 21 September 2011

Image: Roger Halbheer

My Life Story!



Later in
(Michig

May 3, 1971
(Shaker Heig

Appears in
Proceedings Third Annual
ACM Symposium
Theory of Computing
May, 1971

The Complexity of Theorem-Proving Procedures

Stephen A. Cook

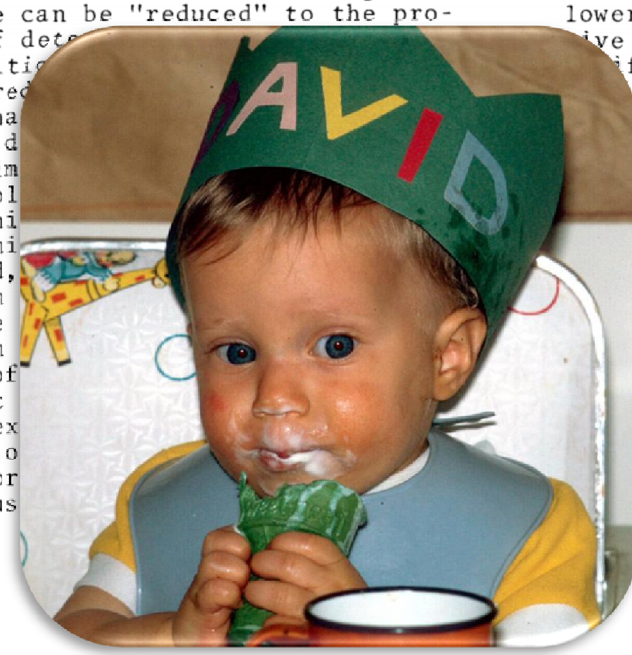
University of Toronto

Summary

It is shown that any recognition problem solved by a polynomial time-bounded nondeterministic Turing machine can be "reduced" to the pro-
blem of det
propositi
Here "re
ing, tha
solved d
mial tim
availabl
From thi
polynomi
defined,
problem
has the
problem
first of
morphic
Other ex
method o
proof pr
calculus

certain recursive set of strings on this alphabet, and we are interested in the problem of finding a good lower bound on its possible recognition times. We provide no such lower bound here, but theorem 1 will give eviden
ifficu
appa
e r
yho
ly
ood
n "
be
der
tro
ike
].

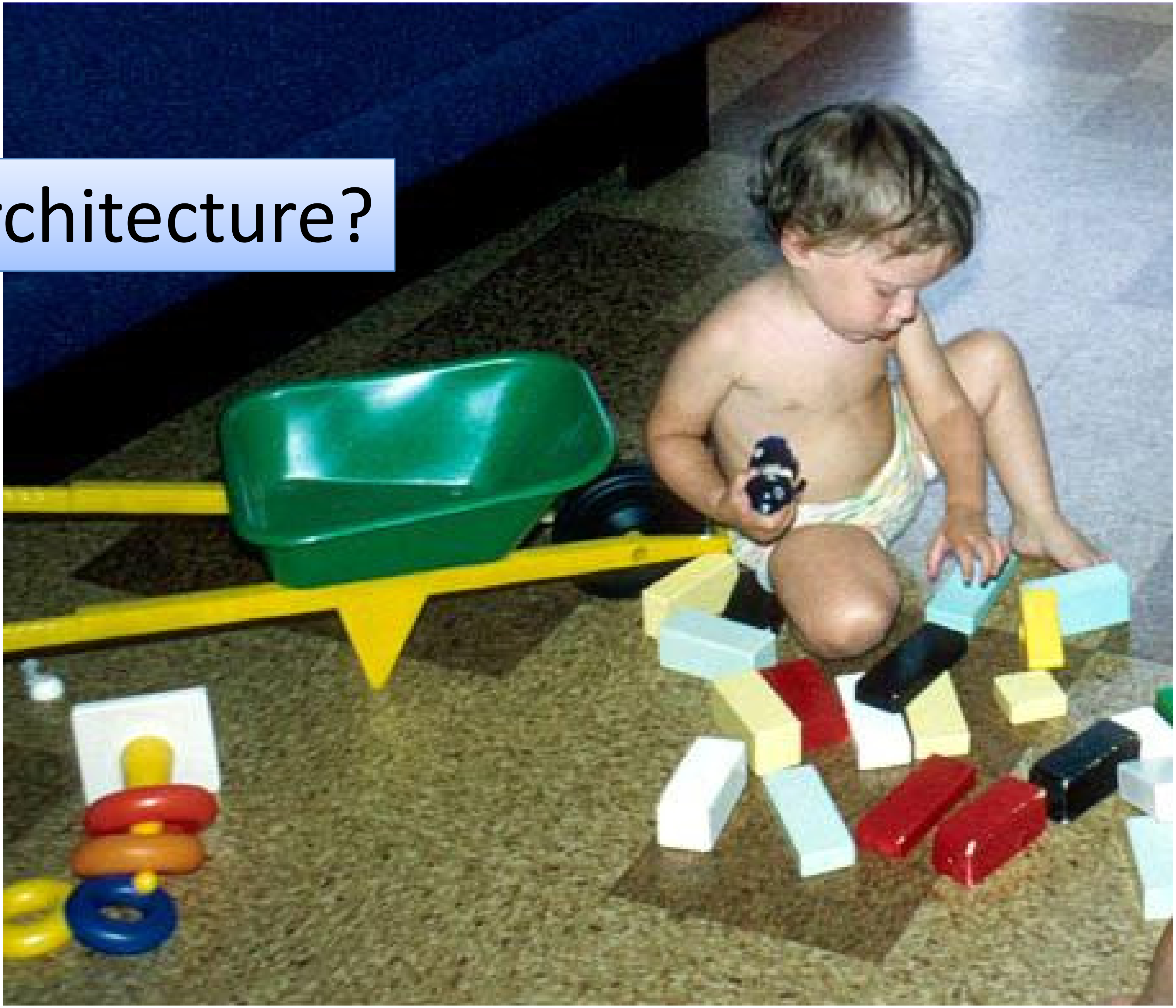
A q
g m
cal
di
uer
. r

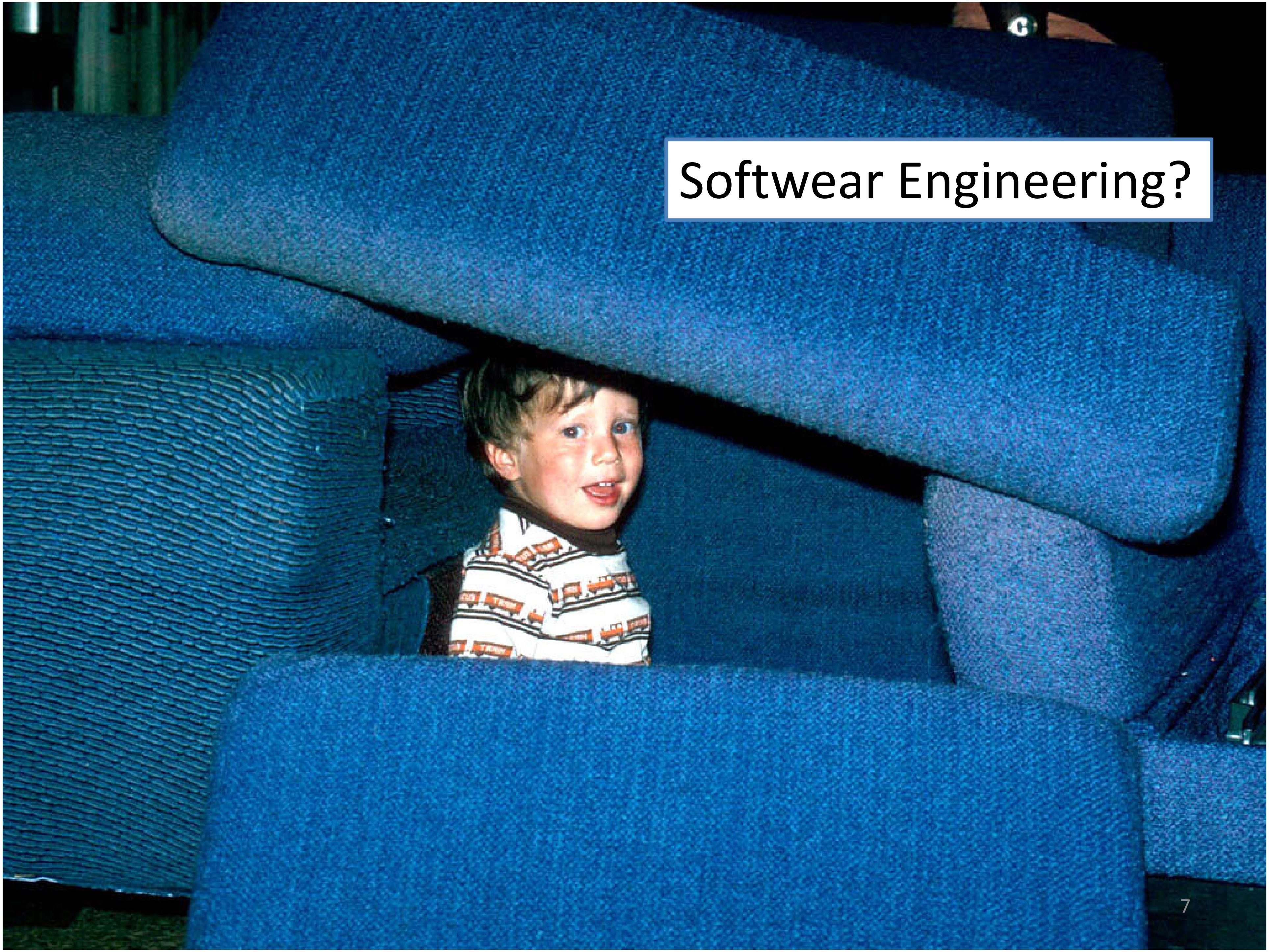


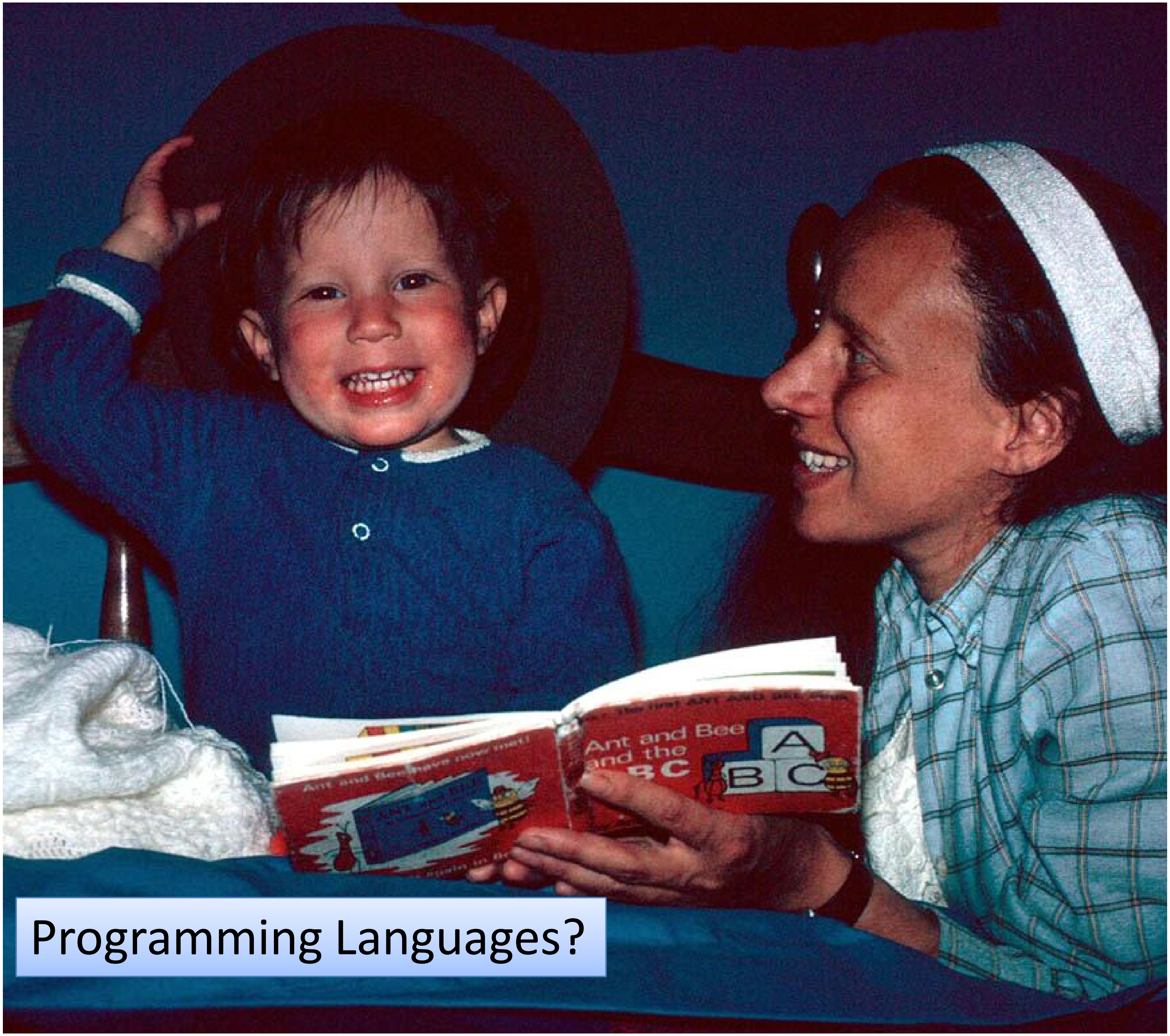
Finding a Research Area


Semiconductor Design?

Architecture?




Softwear Engineering?



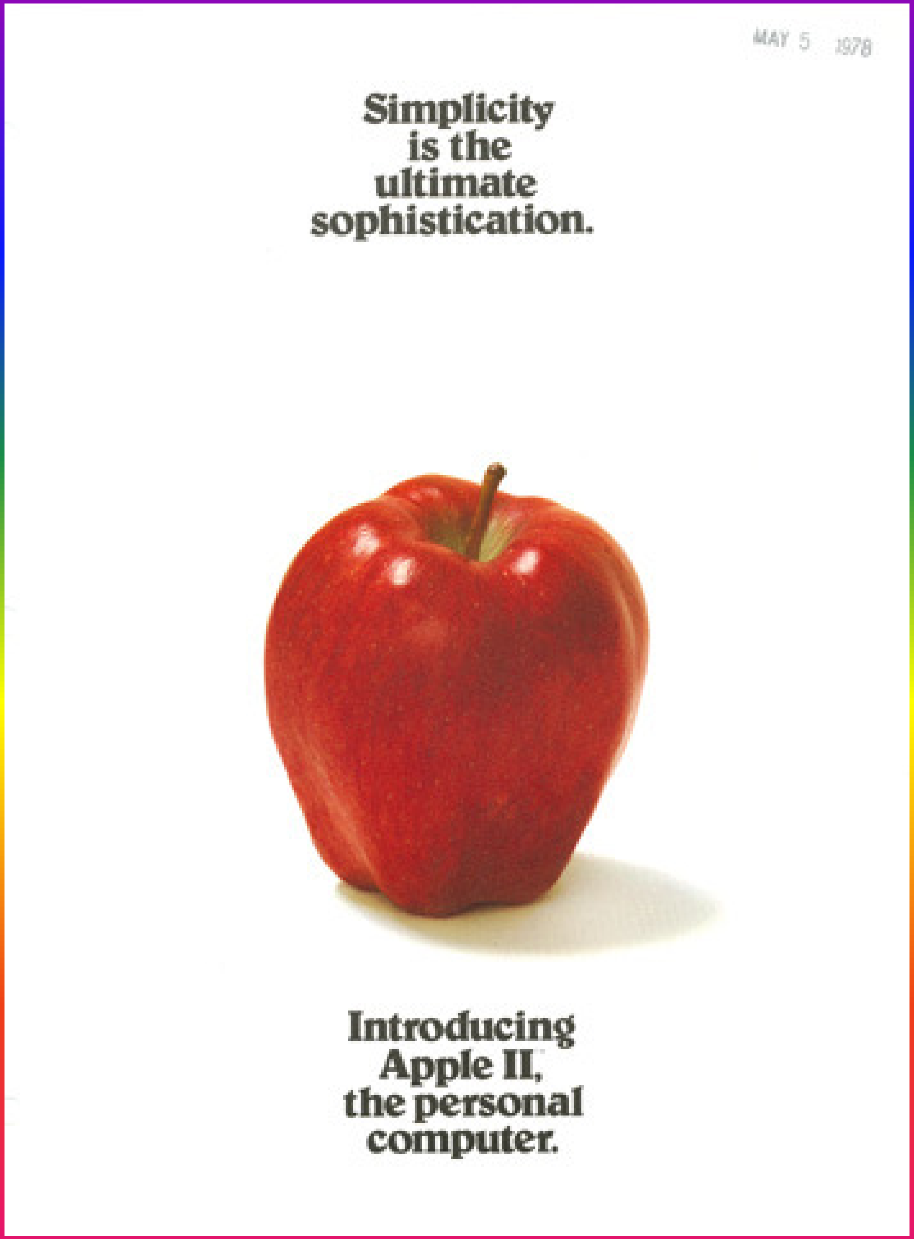
Programming Languages?

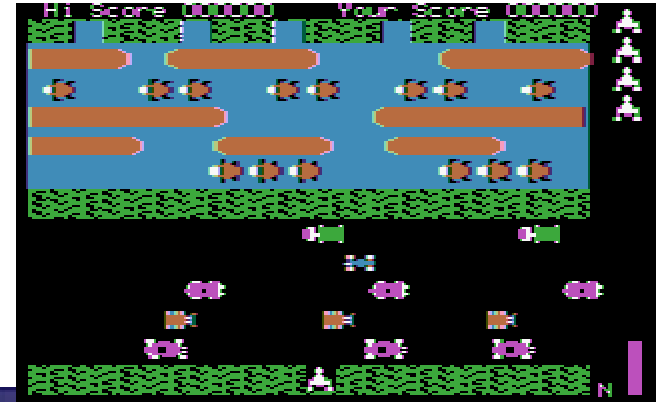

Networking?


Temperature-Aware Computing?


Sensor
Networks?


Information Retrieval?


MAY 5 1978
Simplicity is the ultimate sophistication.
Introducing Apple II, the personal computer.

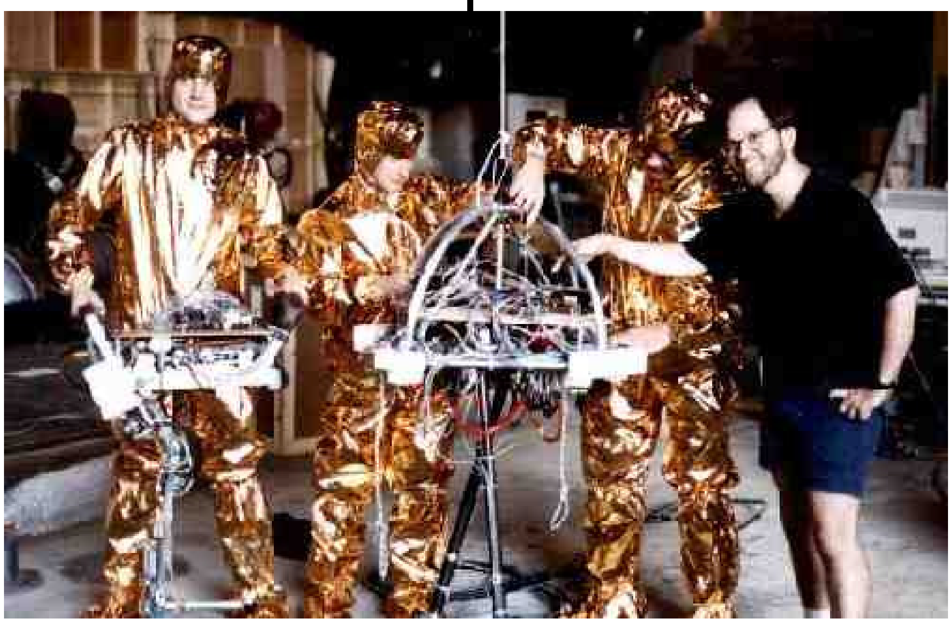


Eureka! Graphics!

1978

1989

1990



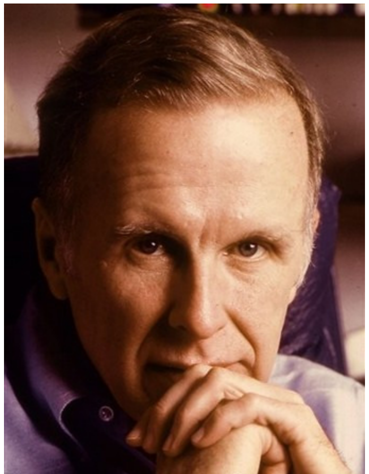
Marc Raibert

David (H.) Evans
(aka, “The Edge”)

David (C.) Evans (and Ivan Sutherland)
Computer Graphics Pioneer

John Backus

(IBM Almaden)



John Guttag (MIT)

mdotm advertising in a post-pc era

Idea for PhD topic

Finished PhD

Joined UVa

Someone noticed I hadn't passed quals yet

1991 1994 1999 2000

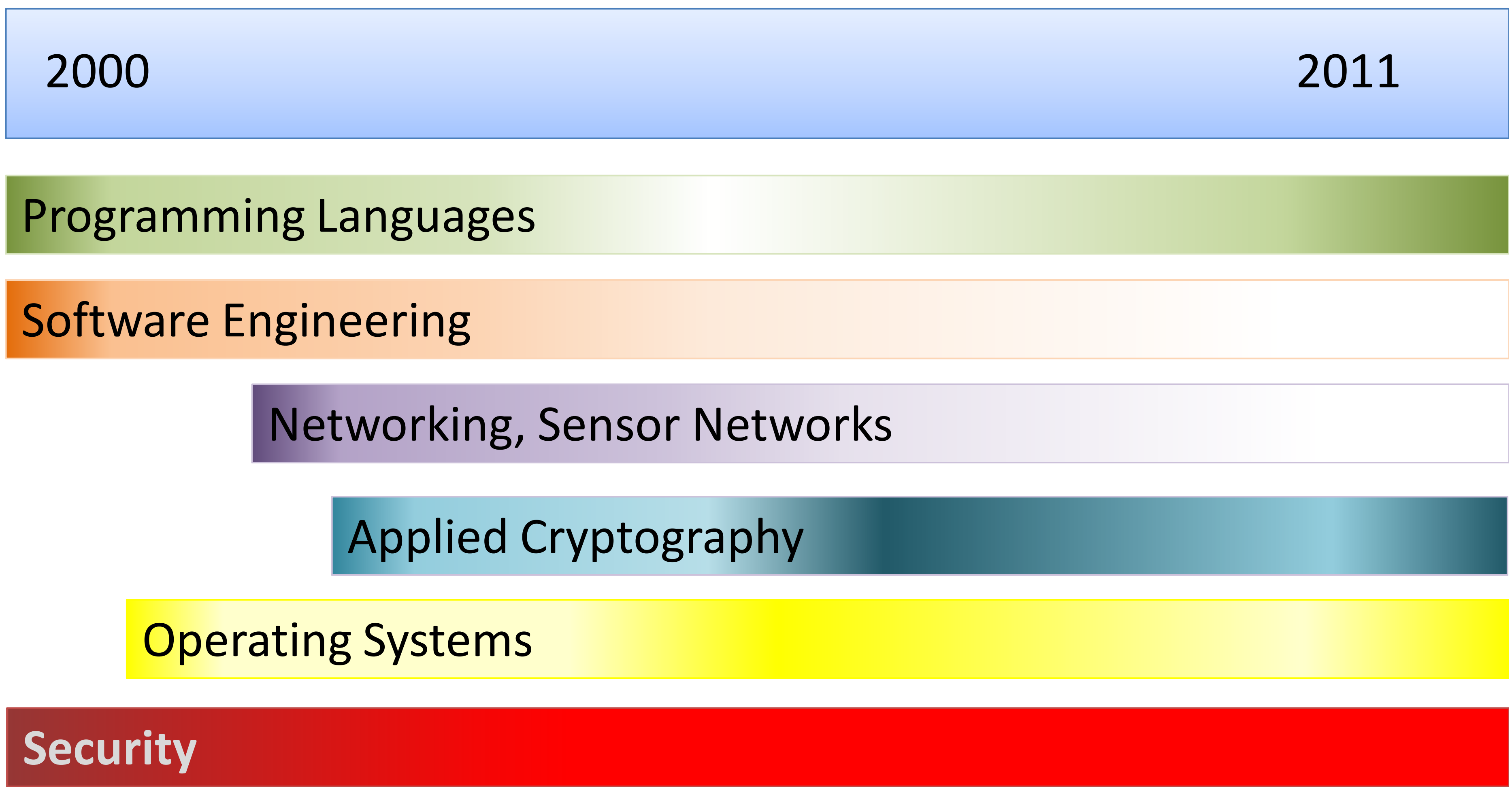
Programming Languages

Software Engineering

splint.org

Security

Naccio


2000
2011
Programming Languages
Software Engineering
Networking, Sensor Networks
Applied Cryptography
Operating Systems
Security

Computer Security

Study of computing systems in the presence of *adversaries*

about what happens when
people don't follow the rules

Security Research Group

At *USENIX Security Symposium*, San Francisco, August 2011

Main Active Projects

DHOSA: End-to-end Security for Web and Smartphone Applications

(MURI with UC Berkeley, Harvard, UIUC, Stonybrook)

Secure Computation (www.securecomputation.org)

(NSF with abhi shelat, UMd, Indiana)

Resilient Clouds

(DARPA with JHU, Purdue)

I am looking for 1-3 students for each of these, but will only talk about Secure Computation today.

Secure Computation with Garbled Circuits


Yan Huang
(UVa Computer Science PhD Student)

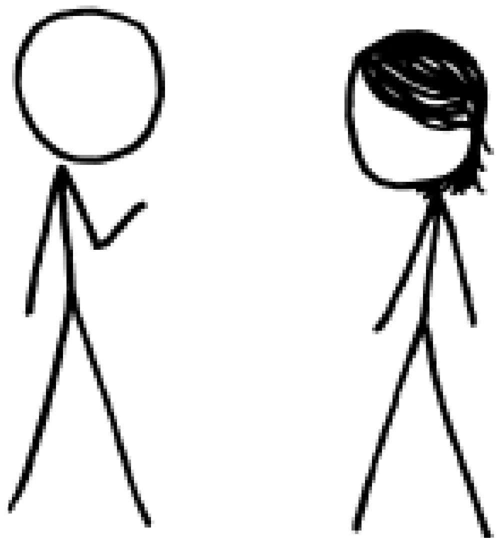

Peter Chapman
(UVa BACS 2012)


Jonathan Katz
(University of Maryland)


Aaron Mackey
(UVa Public Health Genomics)

“Genetic Dating”

Bob



Alice

Genome Compatibility Protocol

WARNING!
Don’t Reproduce

WARNING!
Don’t Reproduce

GenePartner™
Love is no coincidence

23andMe

TheScientist News Current Issue Archive Sur

SHARE

2 comments
Comment on this news story
By Kerry Grens

Forget mistletoe - what about DNA?

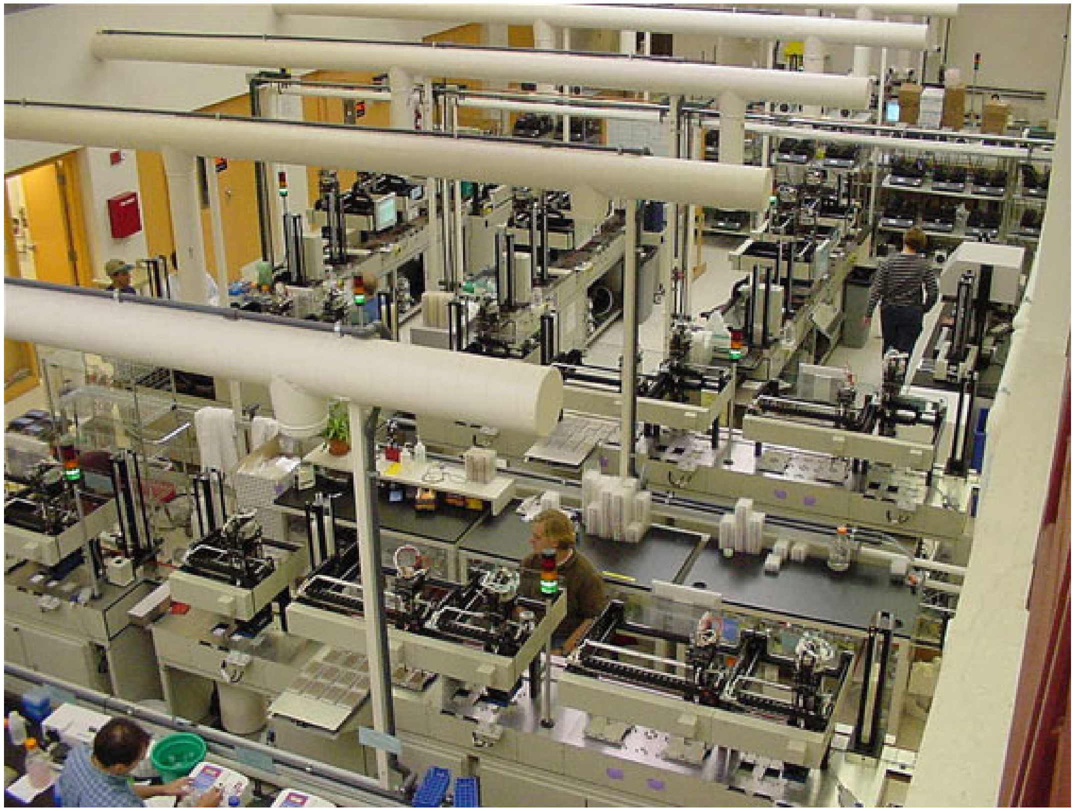
A new dating service matches singles using major histocompatibility complex genes

ScientificMatch.com
"The Science of Love"

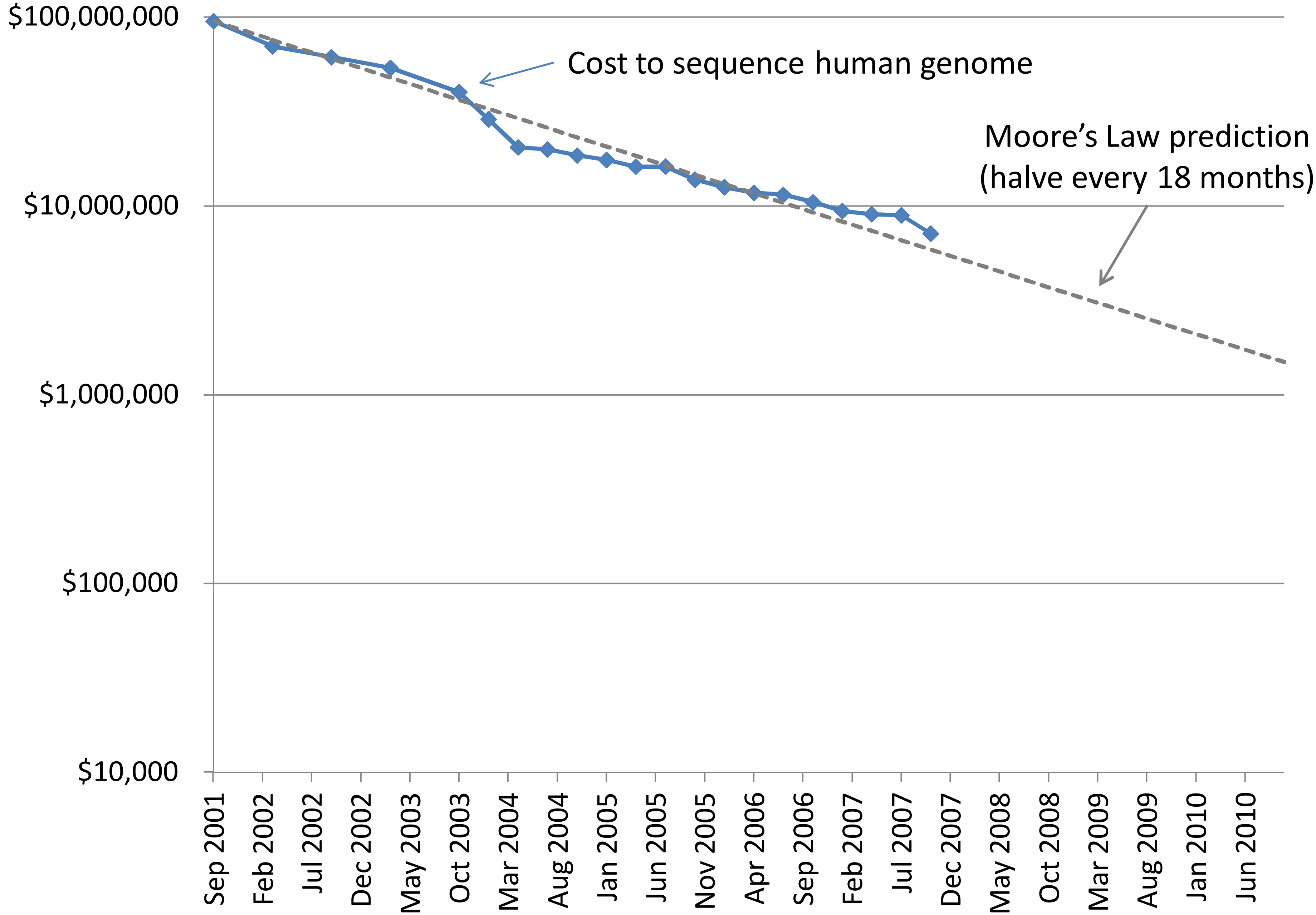
Genome Sequencing

1990: Human Genome Project starts, estimate $3B to sequence one genome ($0.50/base)

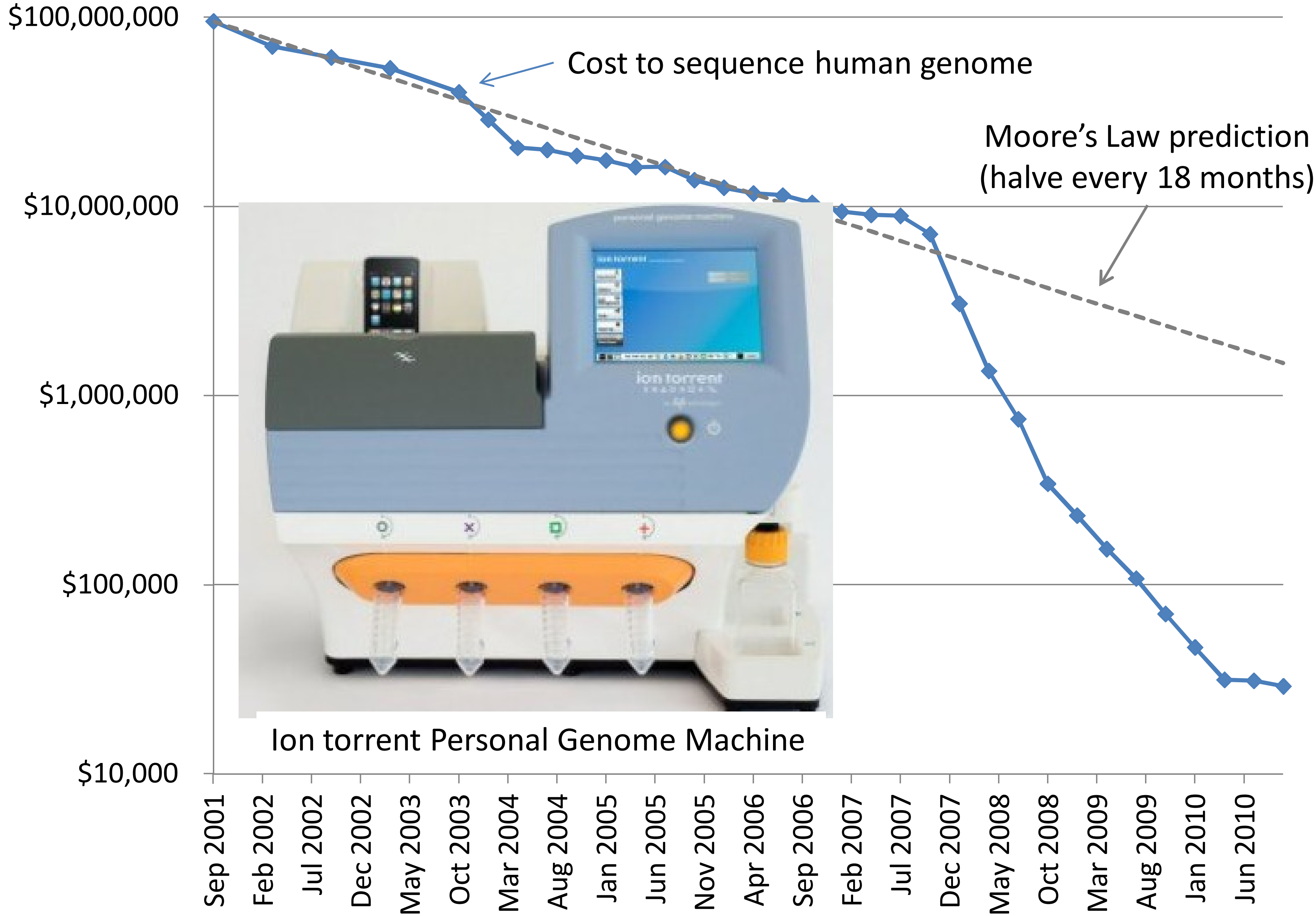
2000: Human Genome Project declared complete, cost ~$300M



Whitehead Institute, MIT



Data from National Human Genome Research Institute: http://www.genome.gov/sequencingcosts

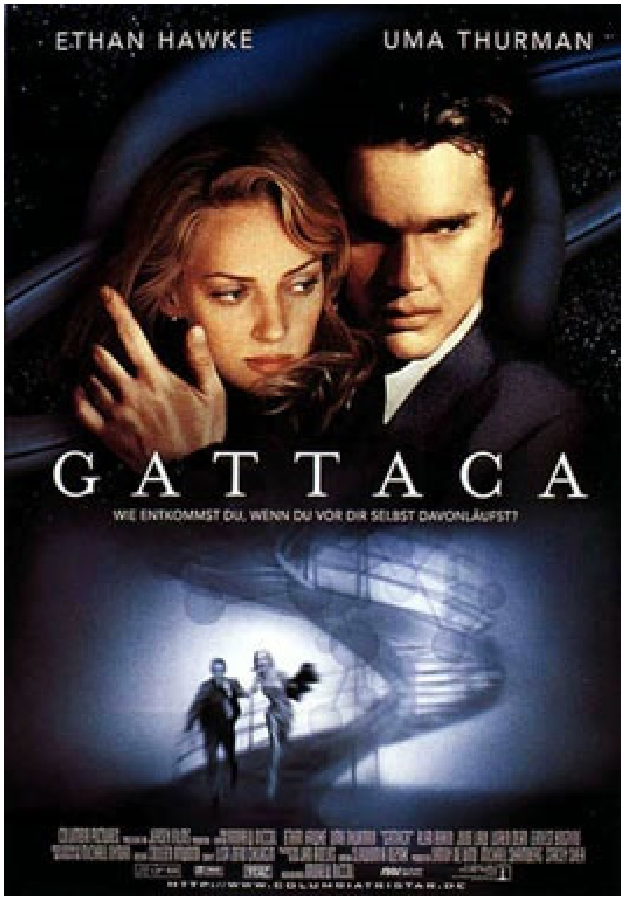


Data from National Human Genome Research Institute: http://www.genome.gov/sequencingcosts

Year	reference	Technology	Sample	Average Reported Coverage depth (fold)	Reported sequencing consumables cost	Estimated cost per 40-fold coverage
[redacted]	*S4*	Sanger (ABI)	JCV	7	$10,000,000	$57,000,000
[redacted]	*S5*	Roche(454)	JDW	7	$1,000,000	$5,700,000
[redacted]	*S6*	Illumina	NA18507	30	$250,000	$330,000
[redacted]	*S7*	Helicos	SRQ	28	$48,000	$69,000
2009	this work	this work	NA07022	87	$8,005	$3,700
2009	this work	this work	NA19240	63	$3,451	$2,200
2009	this work	this work	NA20431	45	$1,726	$1,500

***Human Genome Sequencing Using Unchained Base Reads on Self-Assembling DNA Nanoarrays*.** Radoje Drmanac, Andrew B. Sparks, Matthew J. Callow, Aaron L. Halpern, Norman L. Burns, Bahram G. Kermani, Paolo Carnevali, Igor Nazarenko, Geoffrey B. Nilsen, George Yeung, Fredrik Dahl, Andres Fernandez, Bryan Staker, Krishna P. Pant, Jonathan Baccash, Adam P. Borcherding, Anushka Brownley, Ryan Cedeno, Linsu Chen, Dan Chernikoff, Alex Cheung, Razvan Chirita, Benjamin Curson, Jessica C. Ebert, Coleen R. Hacker, Robert Hartlage, Brian Hauser, Steve Huang, Yuan Jiang, Vitali Karpinchyk, Mark Koenig, Calvin Kong, Tom Landers, Catherine Le, Jia Liu, Celeste E. McBride, Matt Morenzoni, Robert E. Morey, Karl Mutch, Helena Perazich, Kimberly Perry, Brock A. Peters, Joe Peterson, Charit L. Pethiyagoda, Kaliprasad Pothuraju, Claudia Richter, Abraham M. Rosenbaum, Shaunak Roy, Jay Shafto, Uladzislau Sharanhovich, Karen W. Shannon, Conrad G. Sheppy, Michel Sun, Joseph V. Thakuria, Anne Tran, Dylan Vu, Alexander Wait Zaranek, Xiaodi Wu, Snezana Drmanac, Arnold R. Oliphant, William C. Banyai, Bruce Martin, Dennis G. Ballinger, George M. Church, Clifford A. Reid. ***Science*, January 2010.**

Dystopia

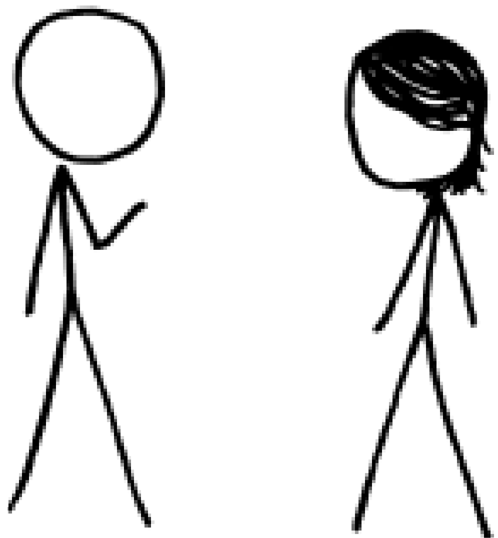


GenePartner™
Love is no coincidence

Personalized Medicine

Secure Two-Party Computation

Bob's Genome: ACTG…
Markers (~1000): [0,1, …, 0]



Alice's Genome: ACTG…
Markers (~1000): [0, 0, …, 1]

Bob

Alice

$$x = f(g_A, g_B)$$

Can Alice and Bob compute a function of their private data, without exposing anything about their data besides the result?

Secure Function Evaluation

Alice (circuit generator)

Bob (circuit evaluator)

Agree on $f(a,b) \to x$

Picks $a \in \{0,1\}^s$

Picks $b \in \{0,1\}^t$

Garbled Circuit Protocol

Outputs $x = f(a,b)$ without revealing a to Bob or b to Alice.

Andrew Yao, 1982/1986

Yao's Garbled Circuits

AND

Inputs		Output
a	*b*	*x*
0 a_0	0 b_0	0 o_0
0 a_0	1 b_1	0 o_0
1 a_1	0 b_0	0 o_0
1 a_1	1 b_1	1 o_1

$Enc_k(m) \to c$

$Enc_k(c) \to m$

$c = Enc_{k=a_0 b_0}(o_0)$

$Enc_{a_0}(o_1)$

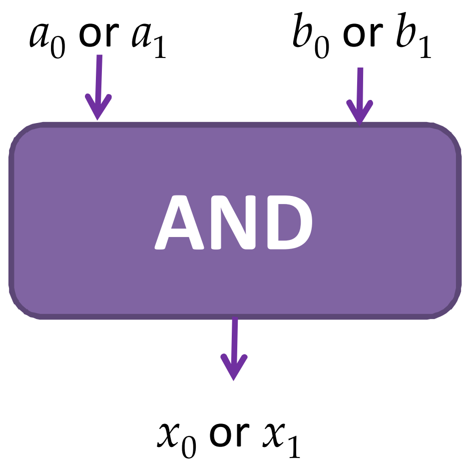
$Enc_{a_1 b_1}(o_1)$

Computing with Garbled Tables

Inputs		Output
a	b	x
a_0	b_0	$Enc_{a0,b0}(x_0)$
a_0	b_1	$Enc_{a0,b1}(x_0)$
a_1	b_0	$Enc_{a1,b0}(x_0)$
a_1	b_1	$Enc_{a1,b1}(x_1)$

Bob can only decrypt **one** of these!

a_i, b_i, x_i are **random** values, chosen by the **circuit generator** but **meaningless** to the **circuit evaluator**.



Garbled And Gate
$Enc_{a0,\ b1}(x_0)$
$Enc_{a1,b1}(x_1)$
$Enc_{a1,b0}(x_0)$
$Enc_{a0,b0}(x_0)$

Garbled Circuit Protocol

Alice (circuit generator)

Creates random keys: $a_0, a_1, b_0, b_1, x_0, x_1$

And Gate
$Enc_{a0,\ b1}(x_0)$
$Enc_{a1,b1}(x_1)$
$Enc_{a1,b0}(x_0)$
$Enc_{a0,b0}(x_0)$

a_0

Sends a_i to Bob based on her input value

Bob (circuit evaluator)

How does the Bob learn his own input wires?

Primitive: **Oblivious Transfer**

Alice

Knows b_0, b_1

Bob

Picks $i \in \{0, 1\}$

Oblivious Transfer Protocol

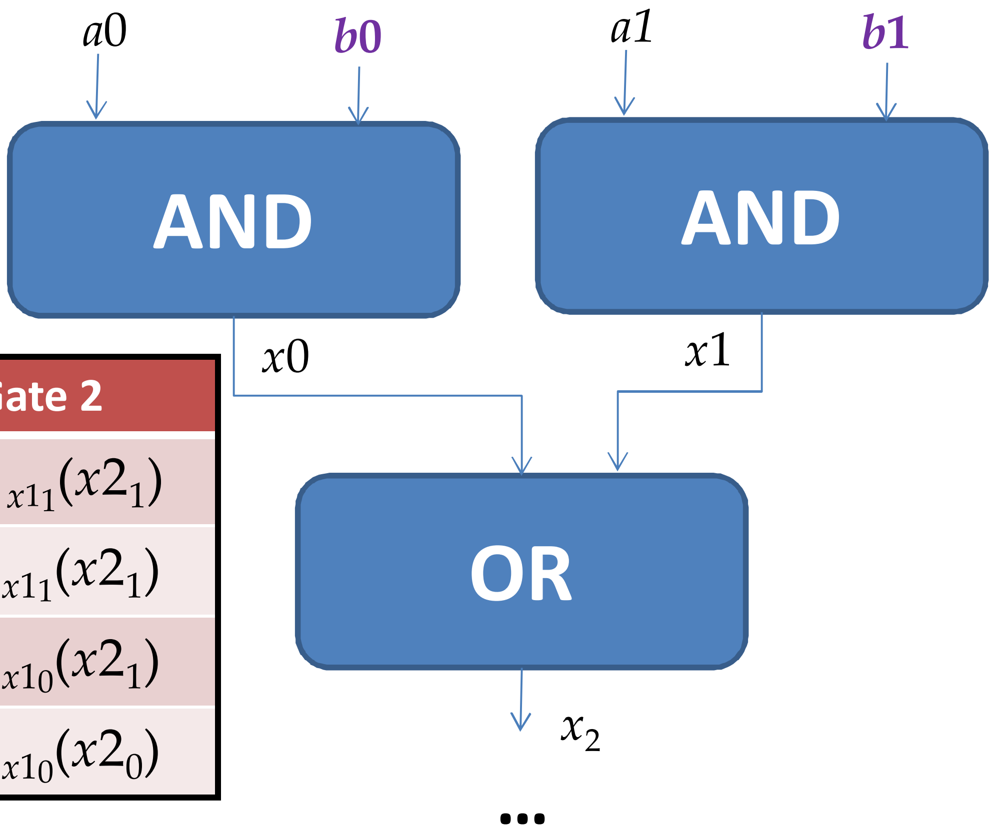
Learns nothing

Learns b_i (only)

Oblivious: Alice doesn't learn which secret Bob obtains
Transfer: Bob learns one of Alice's secrets

Rabin, 1981; Even, Goldreich, and Lempel, 1985; many subsequent papers

Chaining Garbled Circuits

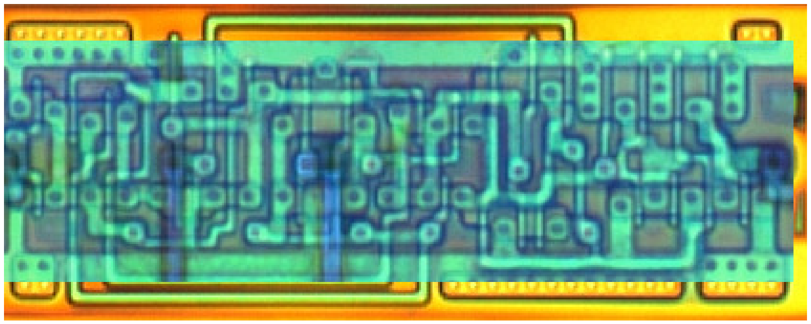


And Gate 1
$Enc_{a1_0,\ b1_1}(x1_0)$
$Enc_{a1_1, b1_1}(x1_1)$
$Enc_{a1_1, b1_0}(x1_0)$
$Enc_{a1_0, b1_0}(x1_0)$

Or Gate 2
$Enc_{x0_0,\ x1_1}(x2_1)$
$Enc_{x0_1, x1_1}(x2_1)$
$Enc_{x0_1, x1_0}(x2_1)$
$Enc_{x0_0, x1_0}(x2_0)$

We can do ***any*** computation privately this way!

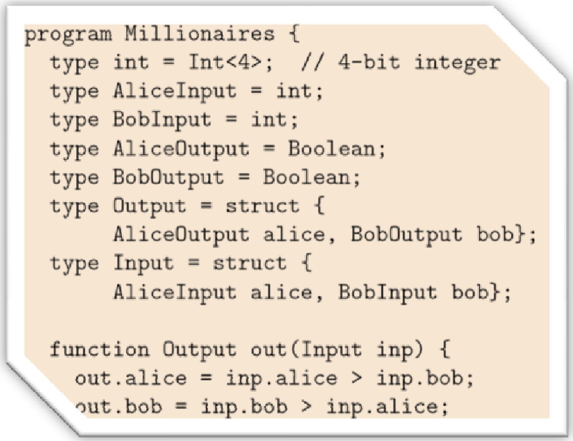
Building Computing Systems



$Enc_{x0_0,\ x1_1}(x2_1)$
$Enc_{x0_1,x1_1}(x2_1)$
$Enc_{x0_1,x1_0}(x2_1)$
$Enc_{x0_0,x1_0}(x2_0)$

Digital Electronic Circuits	Garbled Circuits
Operate on **known data**	Operate on **encrypted wire labels**
One-bit logical operation requires moving a few electrons a few nanometers (hundreds of Billions per second)	One-bit logical operation requires performing (up to) 4 encryption operations (~100,000 gates per second)
Reuse is great!	Reuse is not allowed!
All basic operations have similar cost	Some logical operations “free” (XOR, NOT)

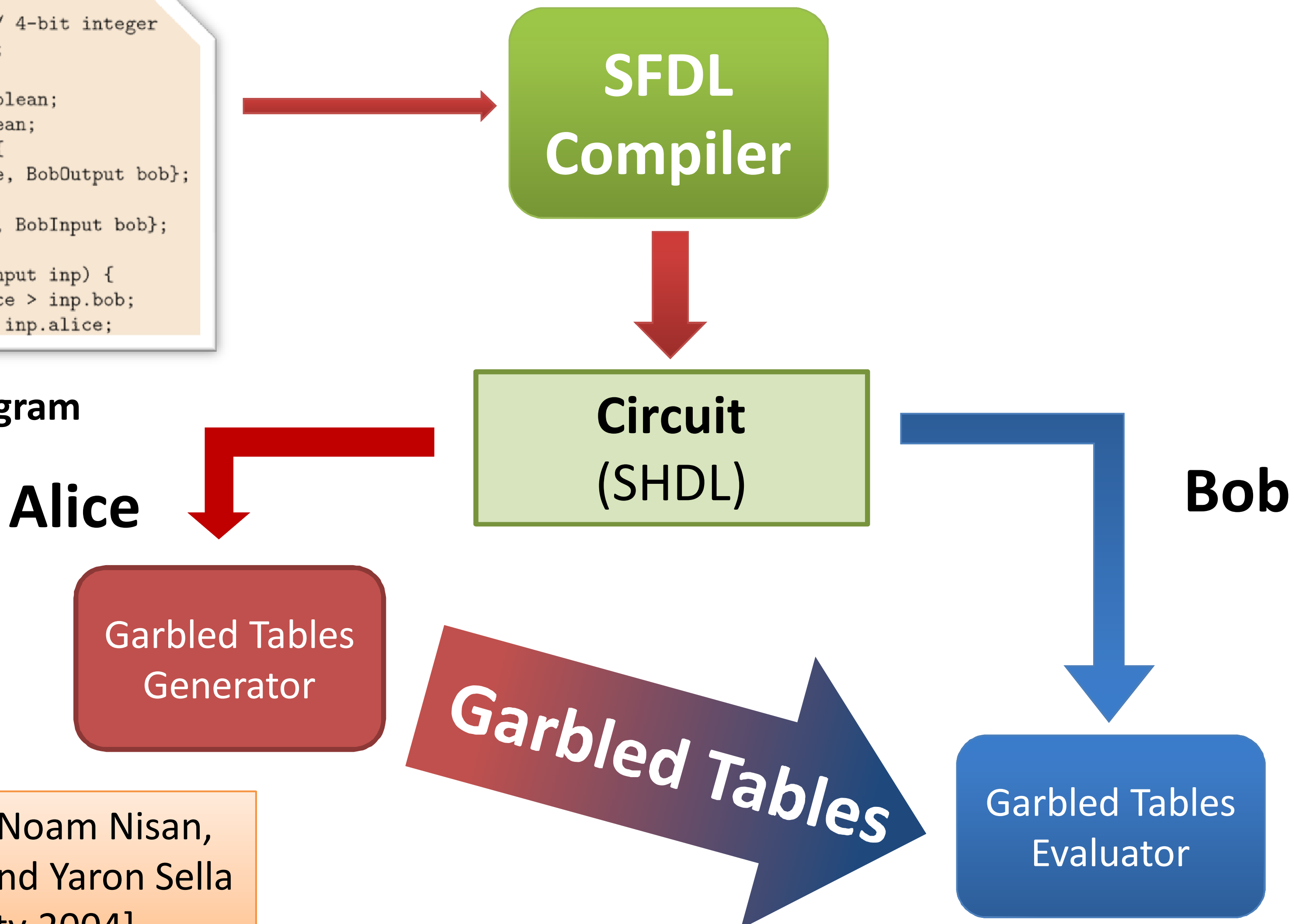
Fairplay



```
program Millionaires {
  type int = Int<4>;  // 4-bit integer
  type AliceInput = int;
  type BobInput = int;
  type AliceOutput = Boolean;
  type BobOutput = Boolean;
  type Output = struct {
       AliceOutput alice, BobOutput bob};
  type Input = struct {
       AliceInput alice, BobInput bob};

  function Output out(Input inp) {
    out.alice = inp.alice > inp.bob;
    out.bob = inp.bob > inp.alice;
```

SFDL Program



Dahlia Malkhi, Noam Nisan, Benny Pinkas and Yaron Sella [USENIX Security 2004]

Problems?

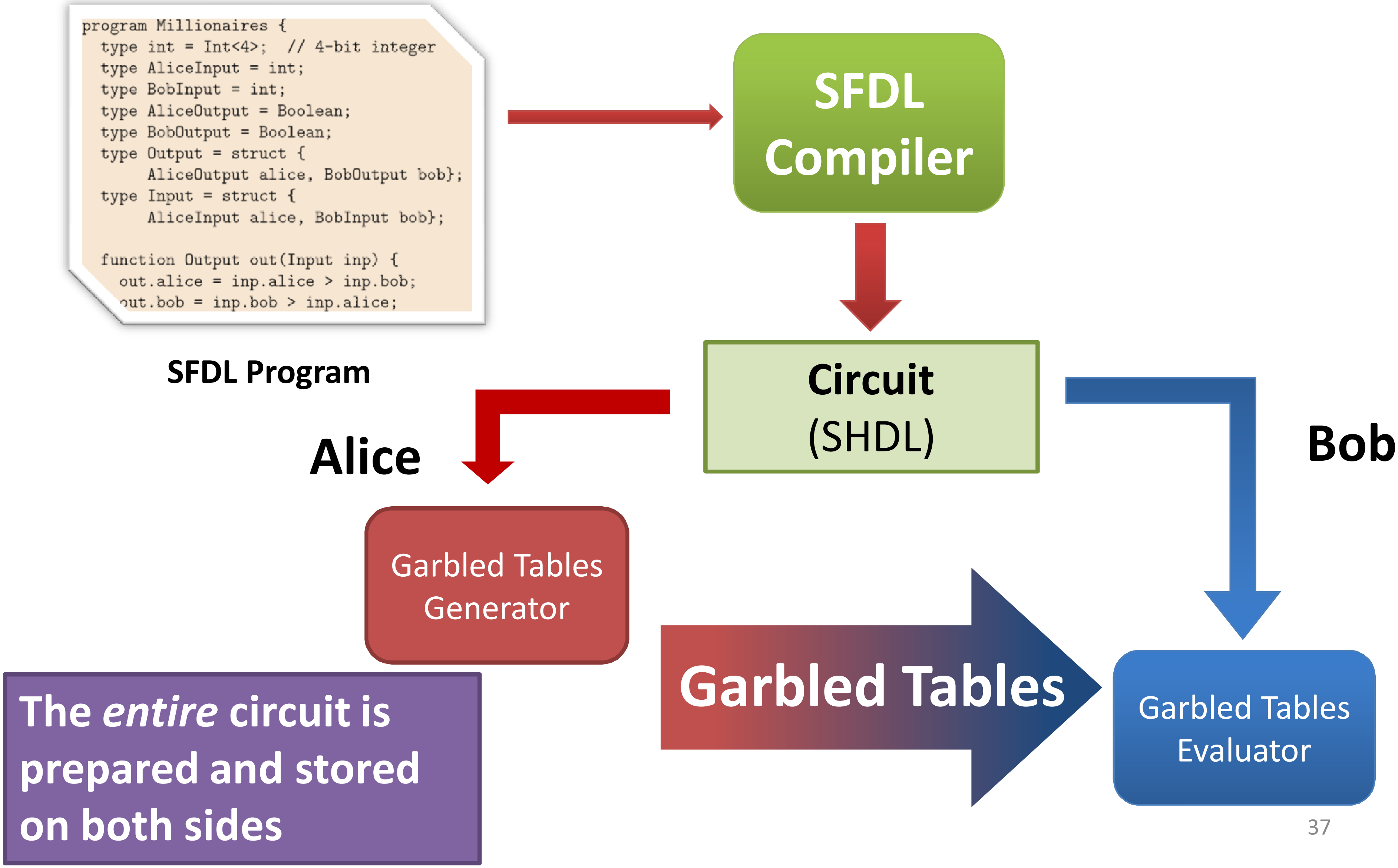
An alternative approach … would have been to apply Yao's generic secure two-party protocol…. This would have required expressing the algorithm as a circuit … and then sending and computing that circuit…. **[We] believe that the performance of our protocols is significantly better than that of applying generic protocols.**

Margarita Osadchy, Benny Pinkas, Ayman Jarrous, Boaz Moskovich.
SCiFI – A System for Secure Face Identification. Oakland 2010.

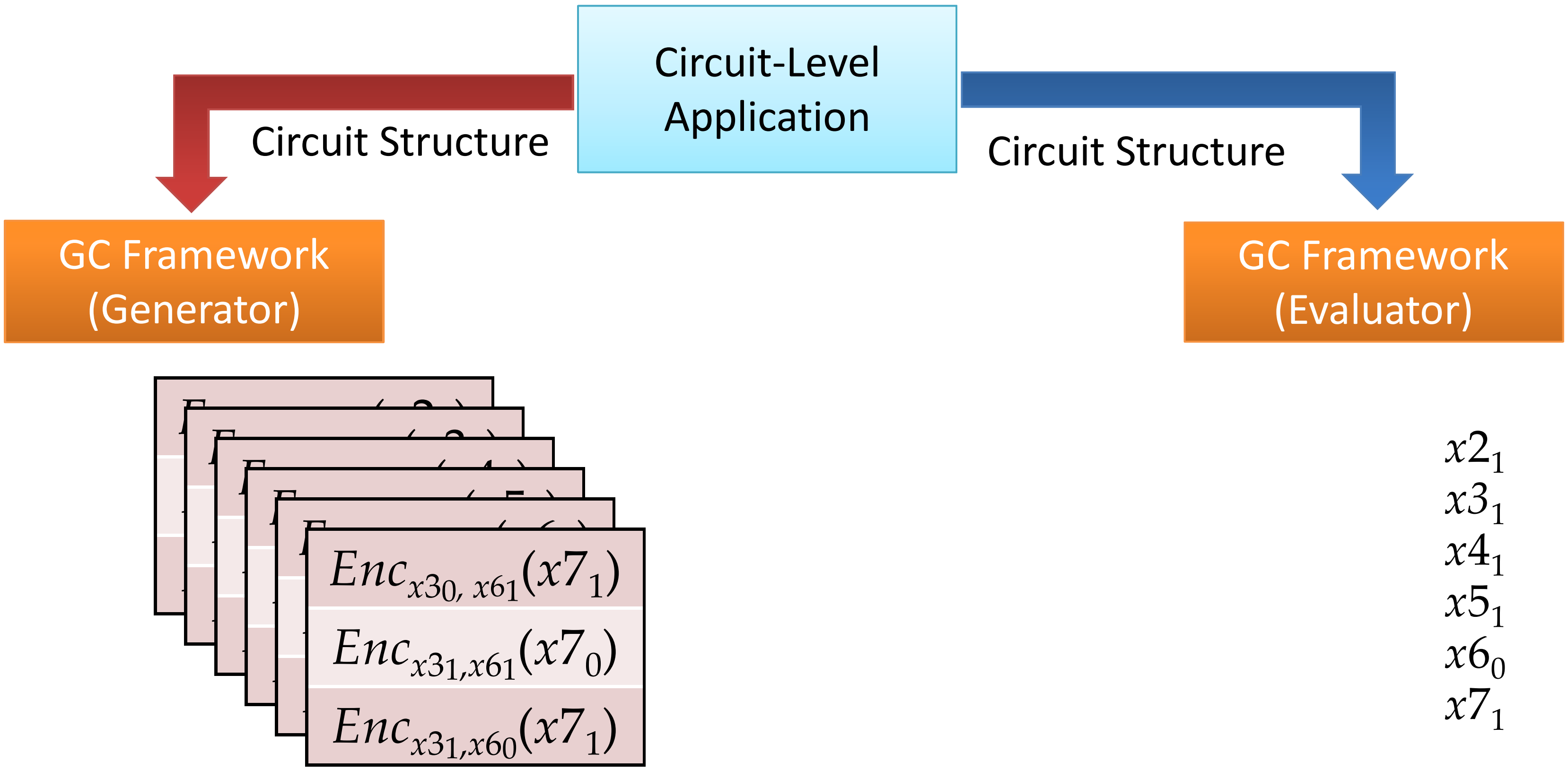
[Generic SFE] is very fast … but the circuit size is extremely large…. Our prototype circuit compiler can compile circuits for problems of size (200, 200) but uses almost 2 GB of memory to do so…. **larger circuits would be constrained by available memory for constructing their garbled versions.**

Somesh Jha, Louis Kruger, Vitaly Shmatikov.
Towards Practical Privacy for Genomic Computation. Oakland 2008.

The Fallacy

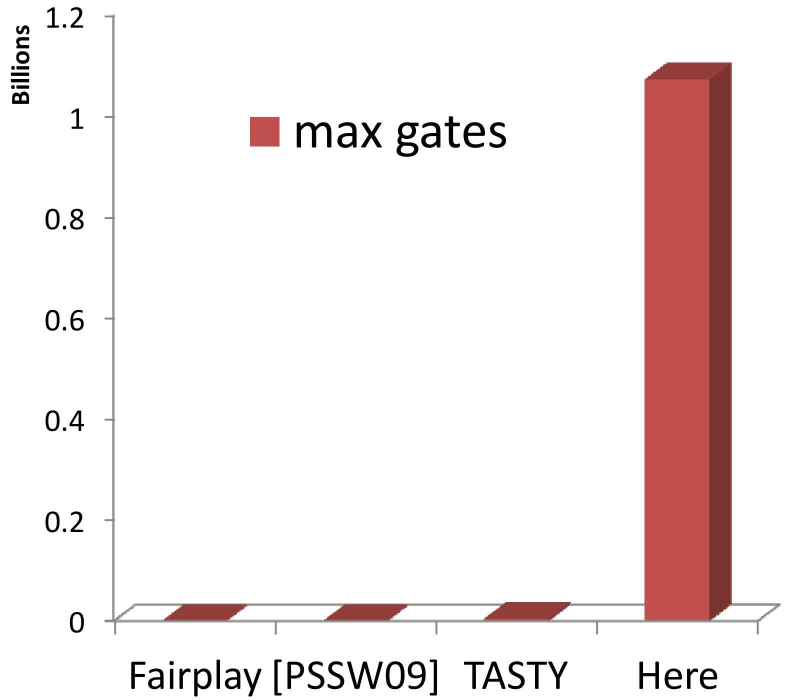

program Millionaires {
type int = Int<4>; // 4-bit integer
type AliceInput = int;
type BobInput = int;
type AliceOutput = Boolean;
type BobOutput = Boolean;
type Output = struct {
AliceOutput alice, BobOutput bob};
type Input = struct {
AliceInput alice, BobInput bob};
function Output out(Input inp) {
out.alice = inp.alice > inp.bob;
out.bob = inp.bob > inp.alice;
SFDL Program
SFDL Compiler
Circuit (SHDL)
Alice
Bob
Garbled Tables Generator
Garbled Tables
Garbled Tables Evaluator
The *entire* circuit is prepared and stored on both sides

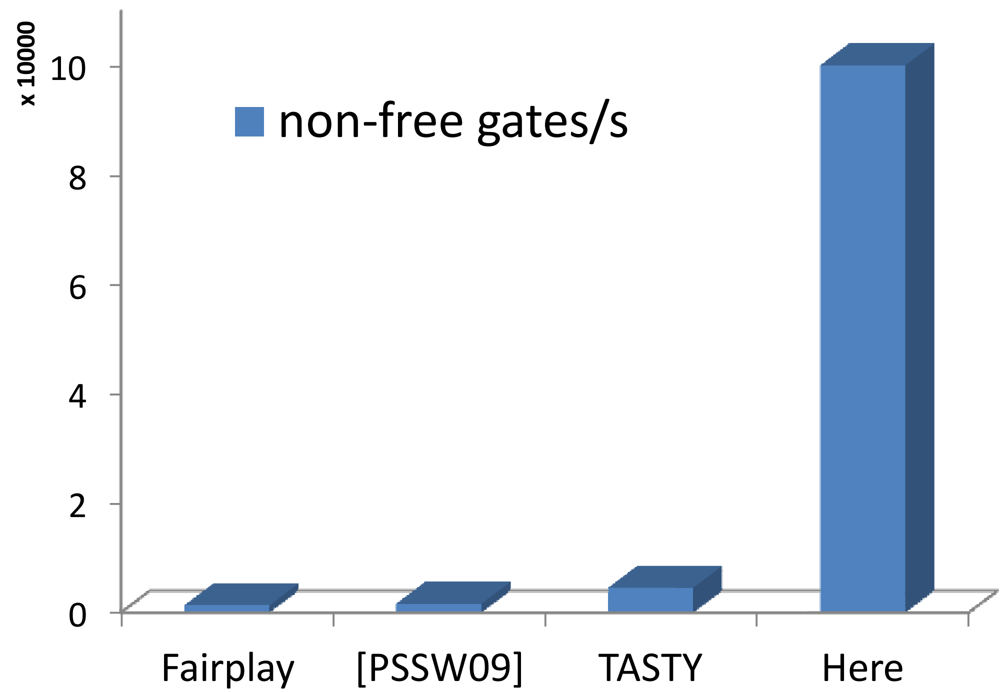
Faster Garbled Circuits



Gates can be evaluated as they are generated: **pipelining**

Results





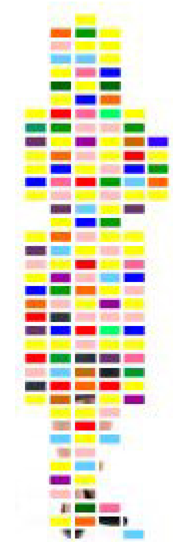
Scalability

Performance

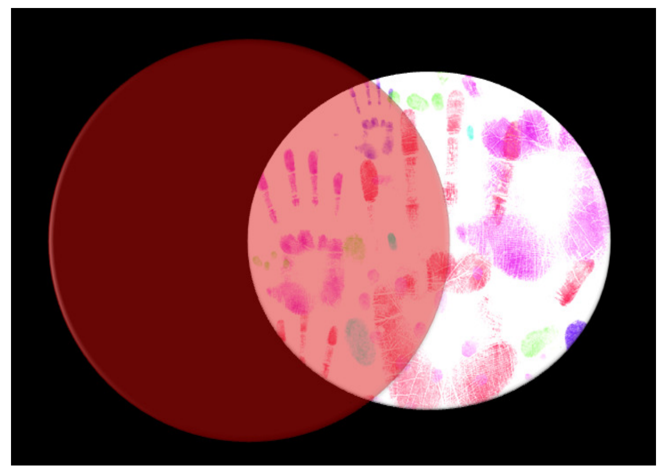
Applications

Privacy-Preserving
Biometric Matching

Private Personal Genomics



Private AES Encryption



Private Set Intersection

Heterozygous Recessive Risk

Counsyl

Alice

Bob		A	a
	A	AA	Aa
	a	aA	aa

carrier (→ Aa)

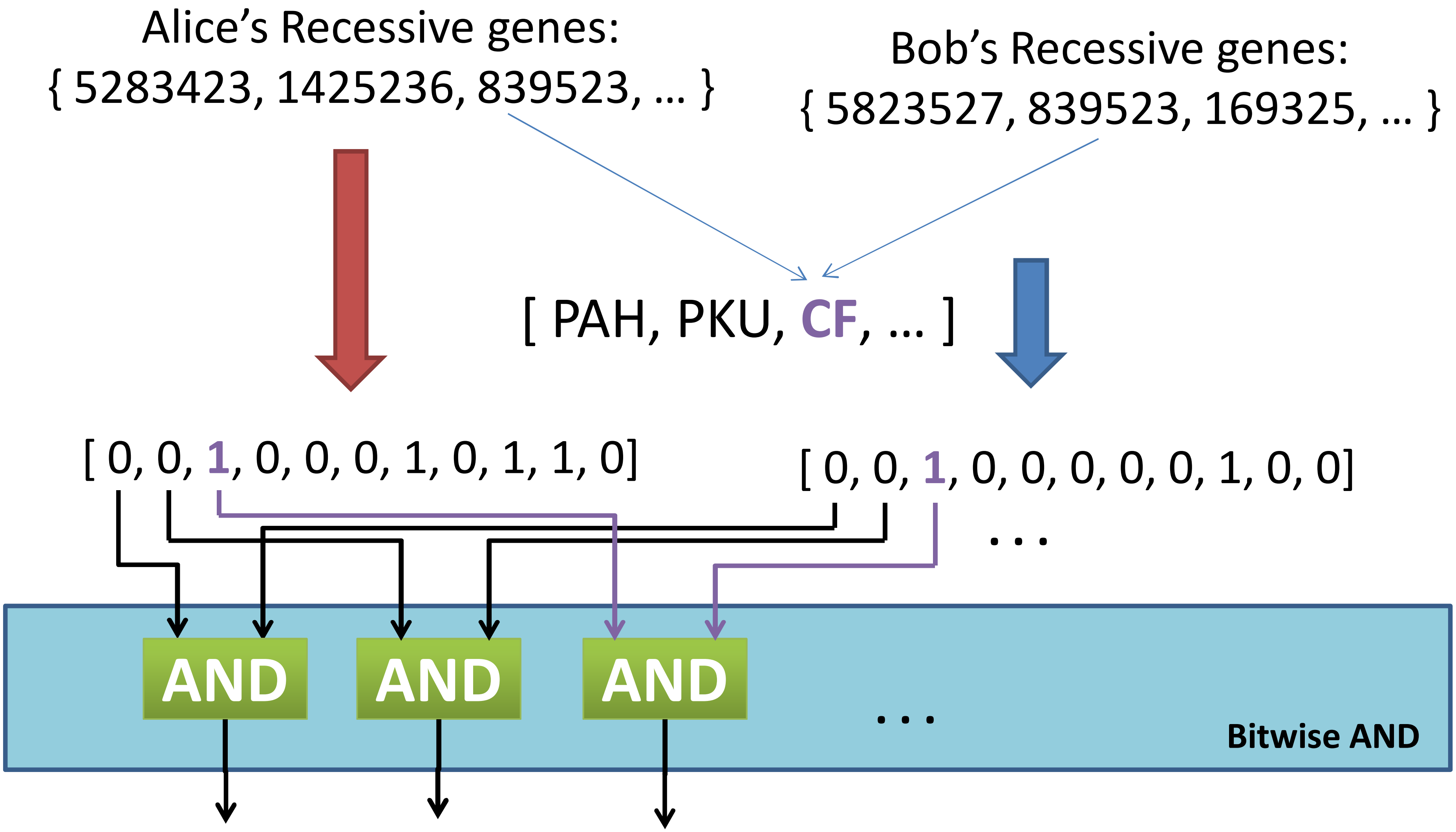
cystic fibrosis (→ aa)

Alice's Heterozygous Recessive genes: { 5283423, 1425236, 839523, … }

Bob's Heterozygous Recessive genes: { 5823527, 839523, 169325, … }

Goal: find the intersection of A and B

Bit Vector Intersection



Scaling

What if there are millions of possible diseases?

Length of bit vector:

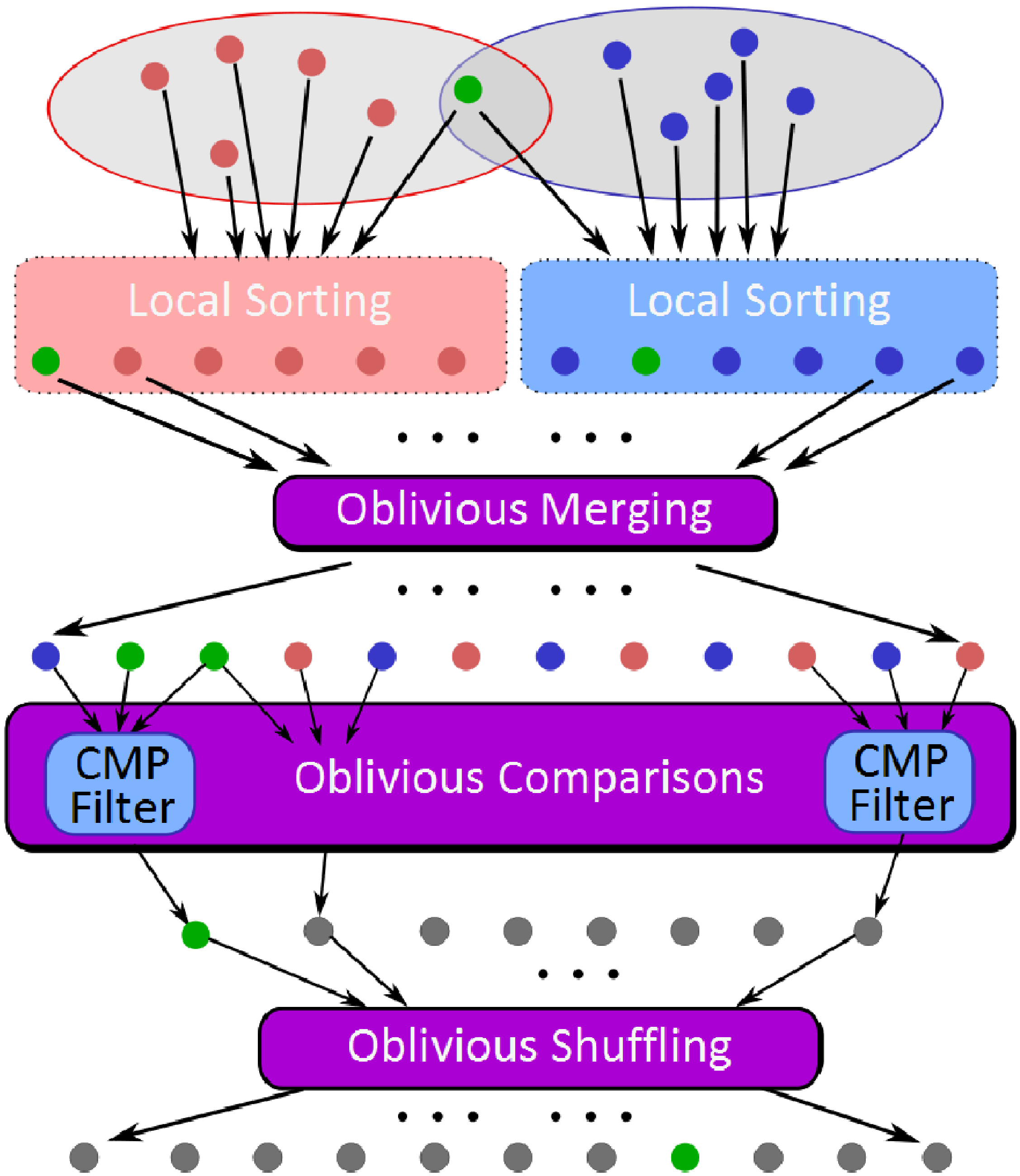
number of possible values

(2^L where L is number of bits for each value)

Other private set intersection problems:

- Do Alice and Bob have any common address book contacts?
- Data mining problems: combine medical records across hospitals
- Two companies want to do joint marketing to common customers

Sort-Compare-Shuffle



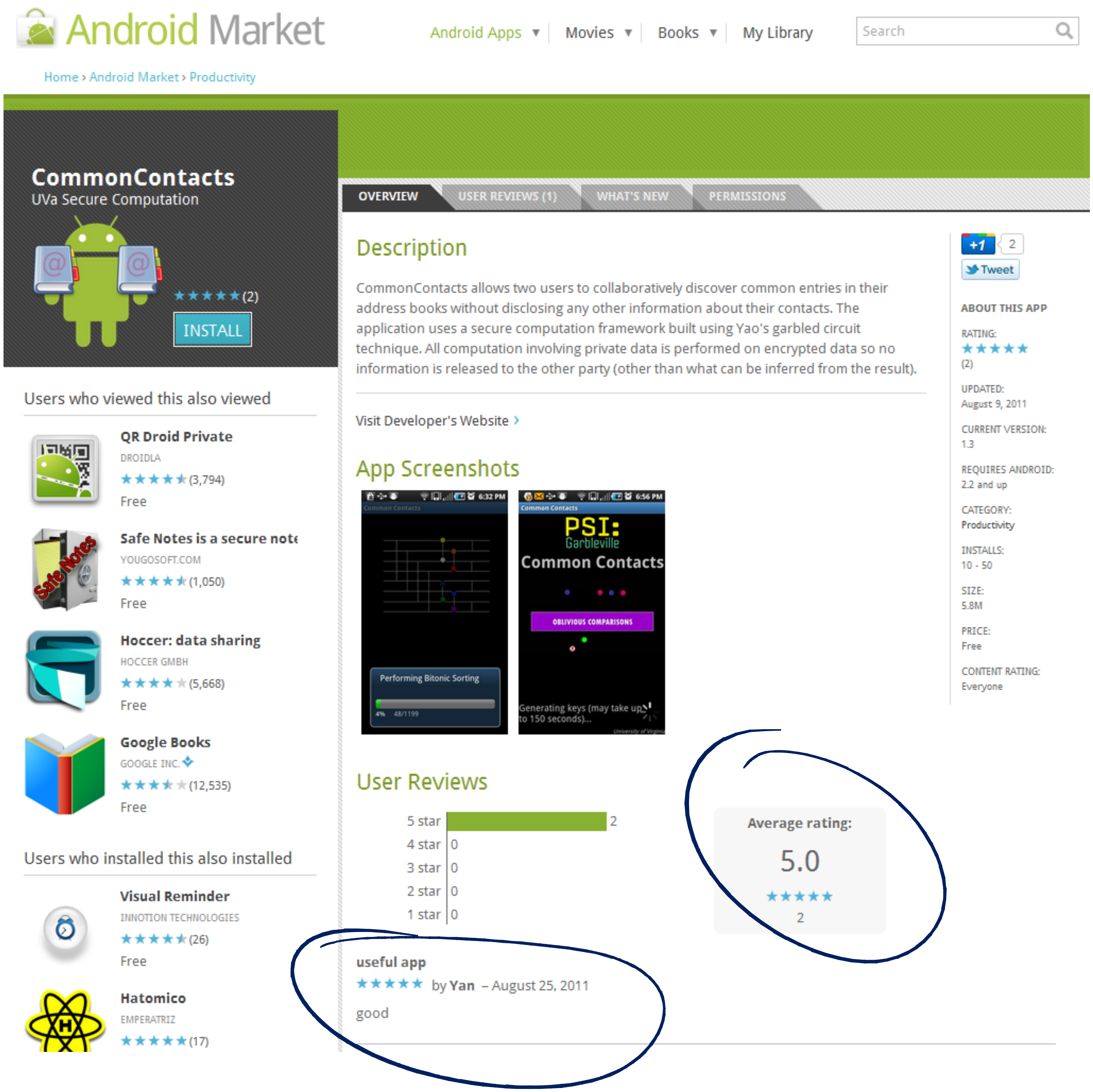
Sort: Take advantage of total order of elements

Compare adjacent elements

Shuffle to hide positions

Venue	Problem	Best Previous Result	Our Result	Speedup
USENIX Security 2011	**Hamming Distance** (Face Recognition, Genetic Dating) – two 900-bit vectors	213s [SCiFI, 2010]	**0.051s**	**4176**
USENIX Security 2011	**Levenshtein Distance** (genome, text comparison) – two 200-character inputs	534s [Jha+, 2008]	**18.4s**	**29**
USENIX Security 2011	**Smith-Waterman** (genome alignment) – two 60-nucleotide sequences	[Not Implementable]	**447s**	**-**
USENIX Security 2011	**AES Encryption**	3.3s [Henecka, 2010]	**0.2s**	**16.5**
NDSS 2011	**Fingerprint Matching** (1024-entry database, 640x8bit vectors)	~83s [Barni, 2010]	**18s**	**4.6**

Scalable: 1 Billion gates evaluated at ~100,000 gates/second on laptop


Android Market
Android Apps
Movies
Books
My Library
Search
Home › Android Market › Productivity
CommonContacts
UVa Secure Computation
(2)
INSTALL
OVERVIEW
USER REVIEWS (1)
WHAT'S NEW
PERMISSIONS
Description
CommonContacts allows two users to collaboratively discover common entries in their address books without disclosing any other information about their contacts. The application uses a secure computation framework built using Yao's garbled circuit technique. All computation involving private data is performed on encrypted data so no information is released to the other party (other than what can be inferred from the result).
Visit Developer's Website ›
App Screenshots
Performing Bitonic Sorting
PSI:
Garbleville
Common Contacts
OBLIVIOUS COMPARISONS
Generating keys (may take up to 150 seconds)...
User Reviews
5 star 2
4 star 0
3 star 0
2 star 0
1 star 0
Average rating:
5.0
2
useful app
by Yan – August 25, 2011
good
+1 2
Tweet
ABOUT THIS APP
RATING:
(2)
UPDATED:
August 9, 2011
CURRENT VERSION:
1.3
REQUIRES ANDROID:
2.2 and up
CATEGORY:
Productivity
INSTALLS:
10 - 50
SIZE:
5.8M
PRICE:
Free
CONTENT RATING:
Everyone
Users who viewed this also viewed
QR Droid Private
DROIDLA
(3,794)
Free
Safe Notes is a secure note
YOUGOSOFT.COM
(1,050)
Free
Hoccer: data sharing
HOCCER GMBH
(5,668)
Free
Google Books
GOOGLE INC.
(12,535)
Free
Users who installed this also installed
Visual Reminder
INNOTION TECHNOLOGIES
(26)
Free
Hatomico
EMPERATRIZ
(17)

Current Projects

Genomics Applications (Aaron Mackey)

Taking advantage of **third-party randomness** (Peter Chapman, Yan Huang)

Using **Partial Evaluation** (PL) (Samee Zahur)

Auditing Leakage: when is it safe to reveal result? (Yikan Chen)

Stronger threat model (Yan Huang, students in abhi's group)

What Next?

Visit our research group blog:
  http://www.jeffersonswheel.org

  Project site for today: www.mightbeevil.com

Come to our research group meetings

  Mailing list: http://www.cs.virginia.edu/evans/srg

Read our recent publications:
  http://www.cs.virginia.edu/evans/pubs/

Arrange to meet with me or come by Rice 507