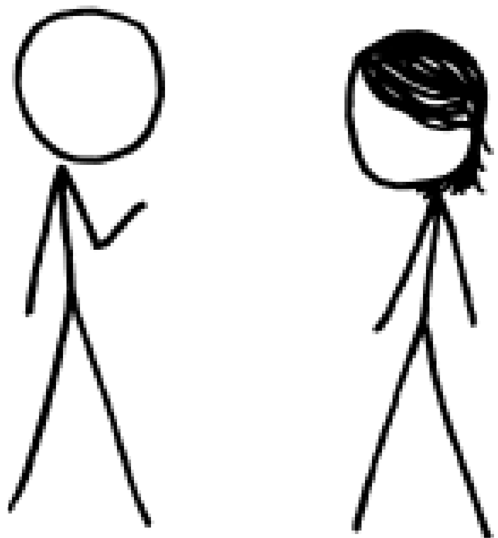
Computing Cooperatively with People You Don't Trust

University of Richmond
30 January 2012

David Evans
University of Virginia
http://www.cs.virginia.edu/evans
http://MightBeEvil.com

“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

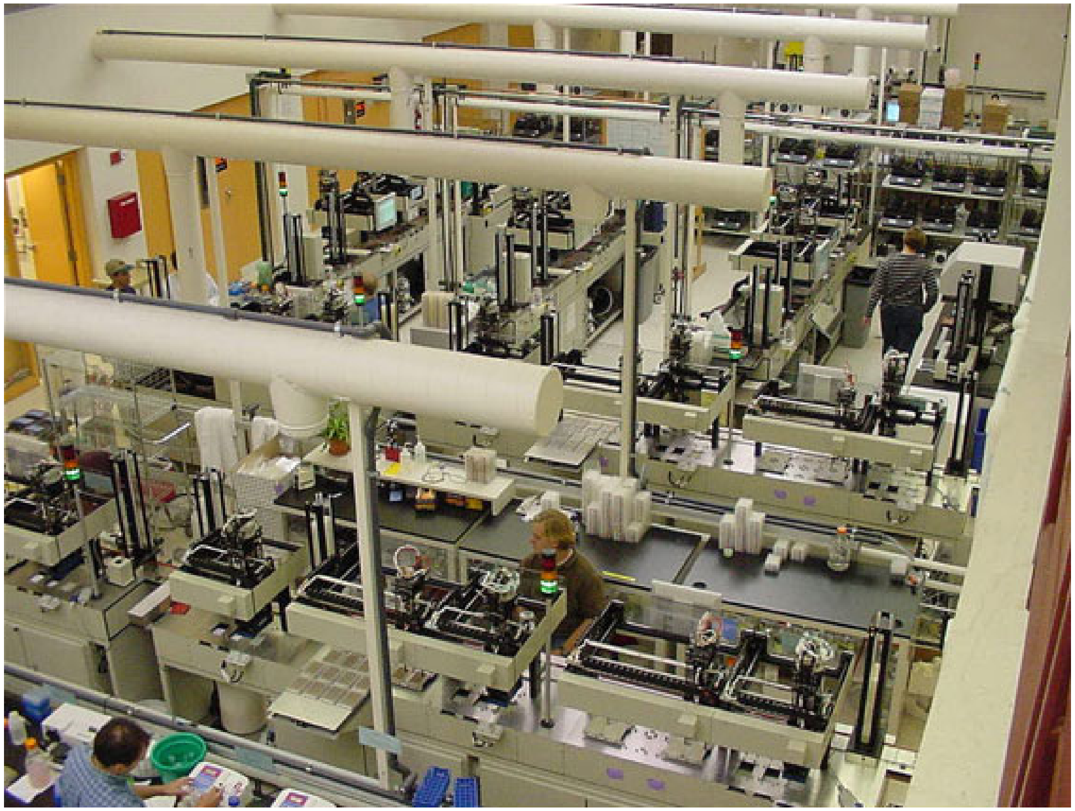
ScientificMatch.com

"The Science of Love"

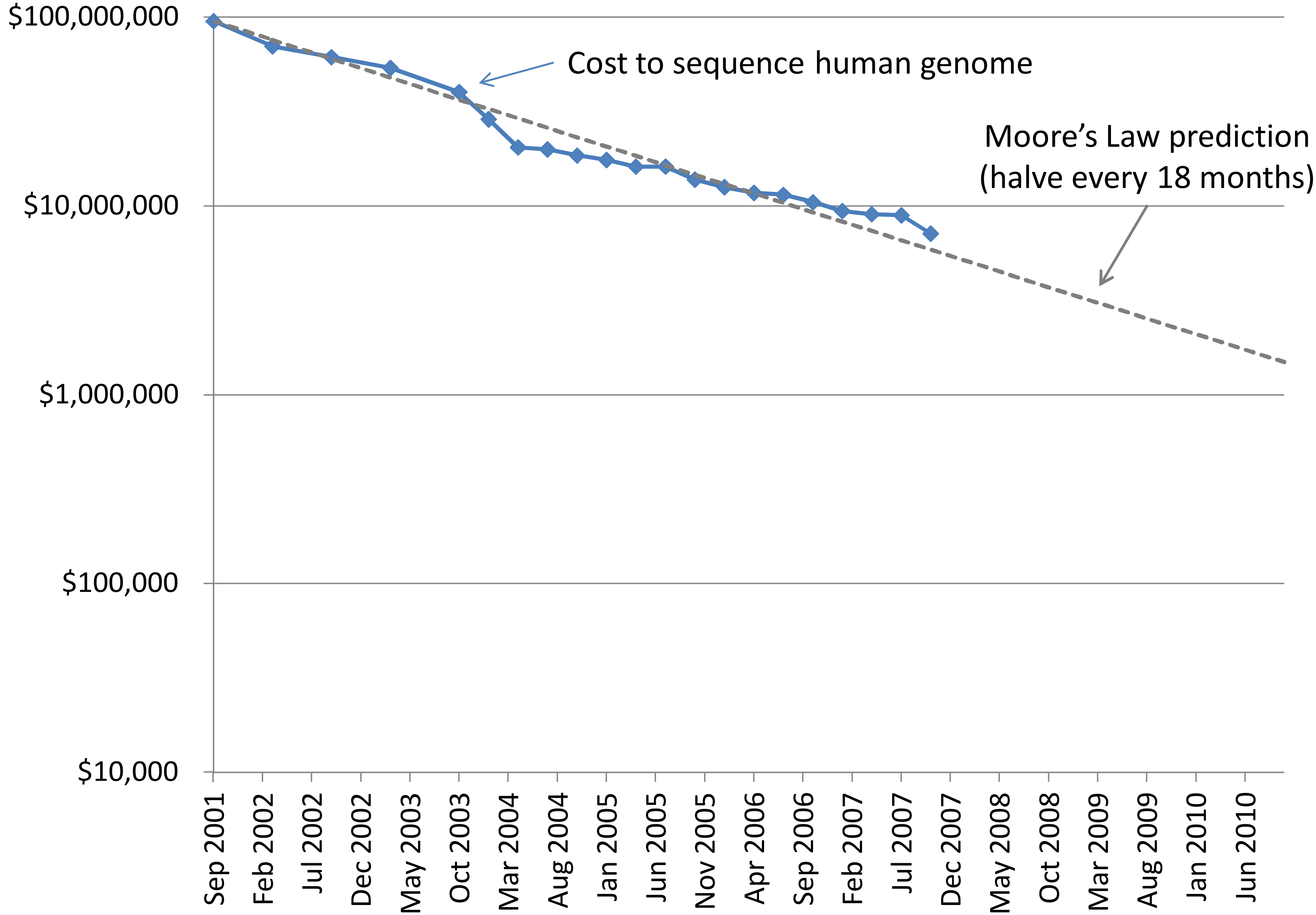
Genome Sequencing

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

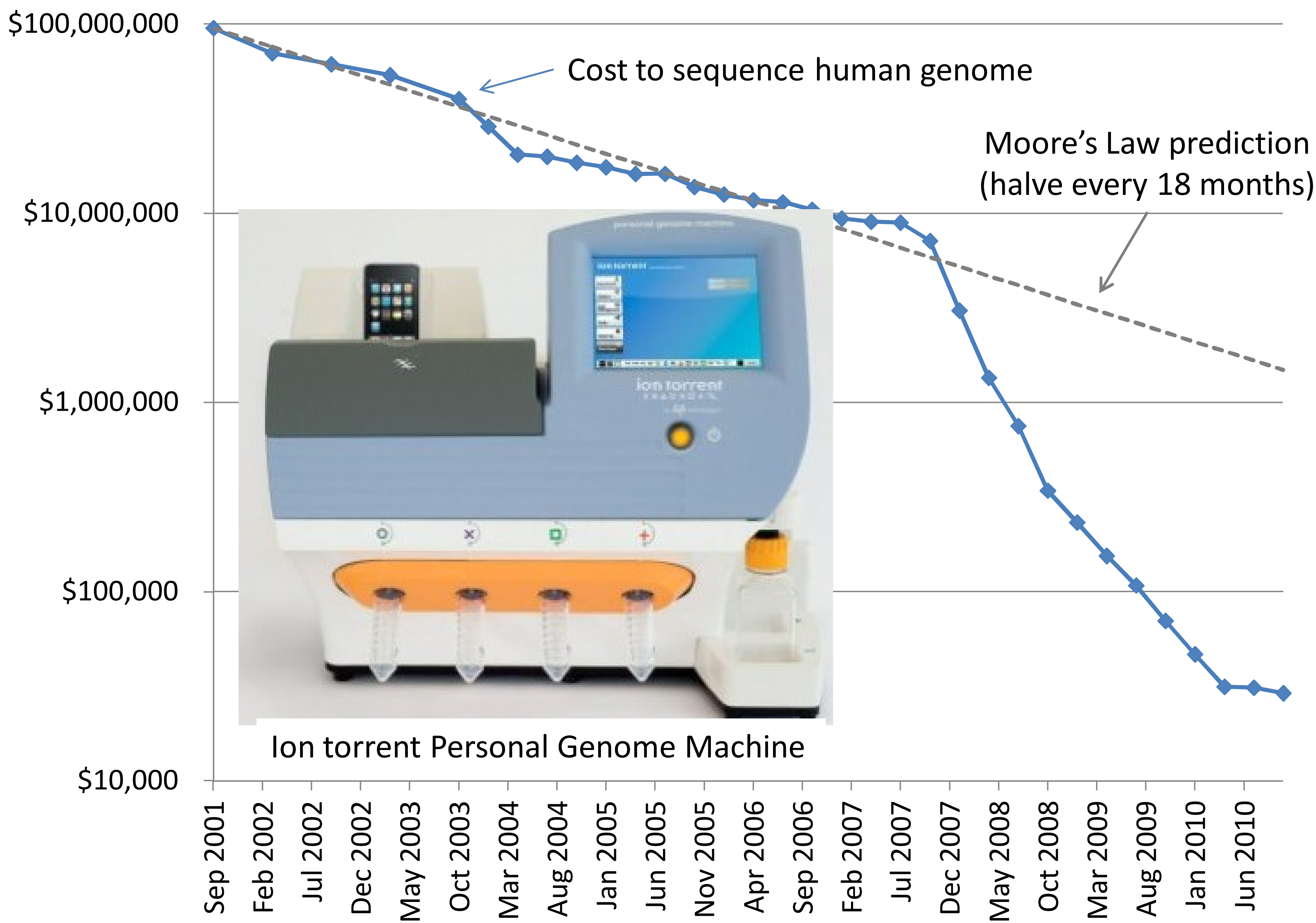
2000: Human Genome Project declared complete, cost ~$300M



Whitehead Institute, MIT



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

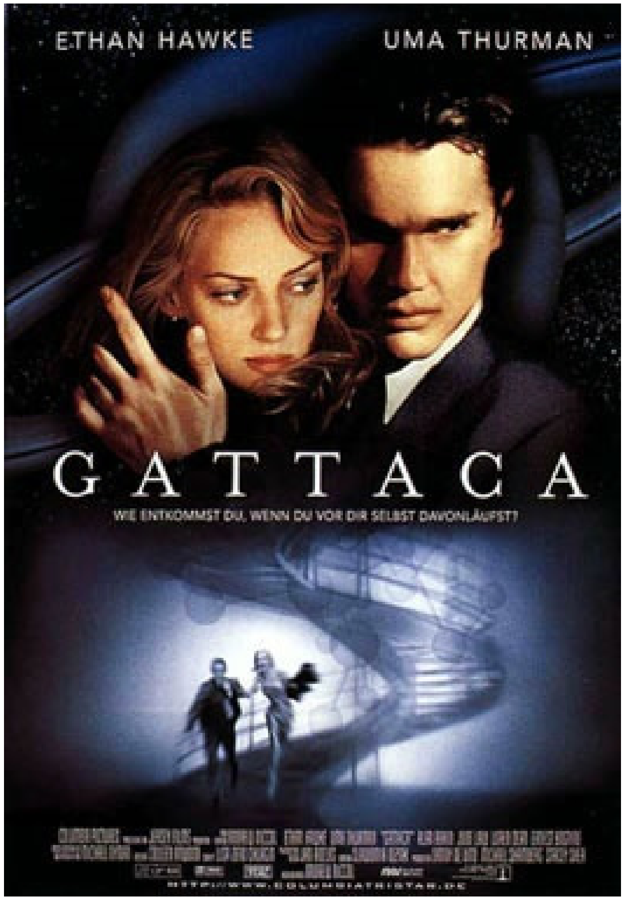


Data from National Human Genome Research Institute: http://www.genome.gov/sequencingcosts

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

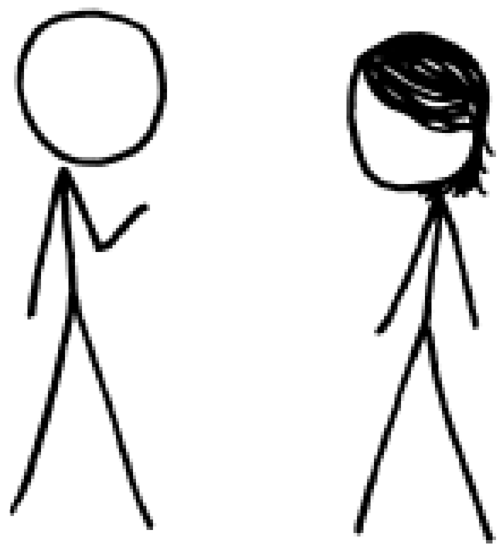
***Human Genome Sequencing Using Unchained Base Reads on Self-Assembling DNA Nanoarrays*.** Radoje Drmanac, Andrew B. Sparks, Matthew J. Callow, Aaron L. Halpern, Norman L. Burns, Bahram G. Kermani, Paolo Carnevali, Igor Nazarenko, Geoffrey B. Nilsen, George Yeung, Fredrik Dahl, Andres Fernandez, Bryan 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



Personalized Medicine

Secure Two-Party Computation



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

Bob

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

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

$f(a,b) \rightarrow x$

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

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

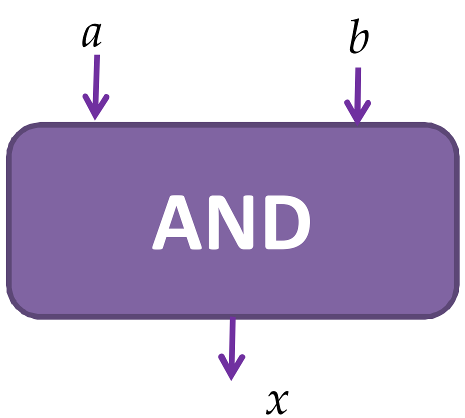
Garbled Circuit Protocol

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

Andrew Yao, 1982/1986

Regular Logic

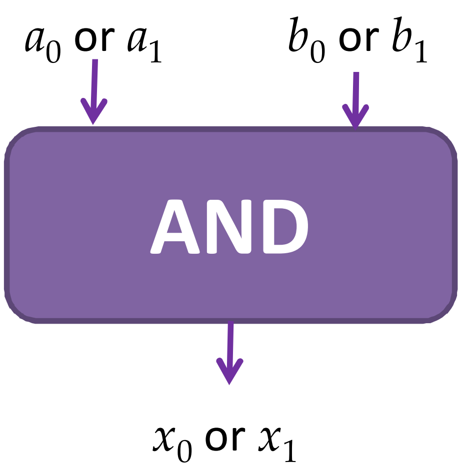
Inputs		Output
a	b	x
0	0	0
0	1	0
1	0	0
1	1	1



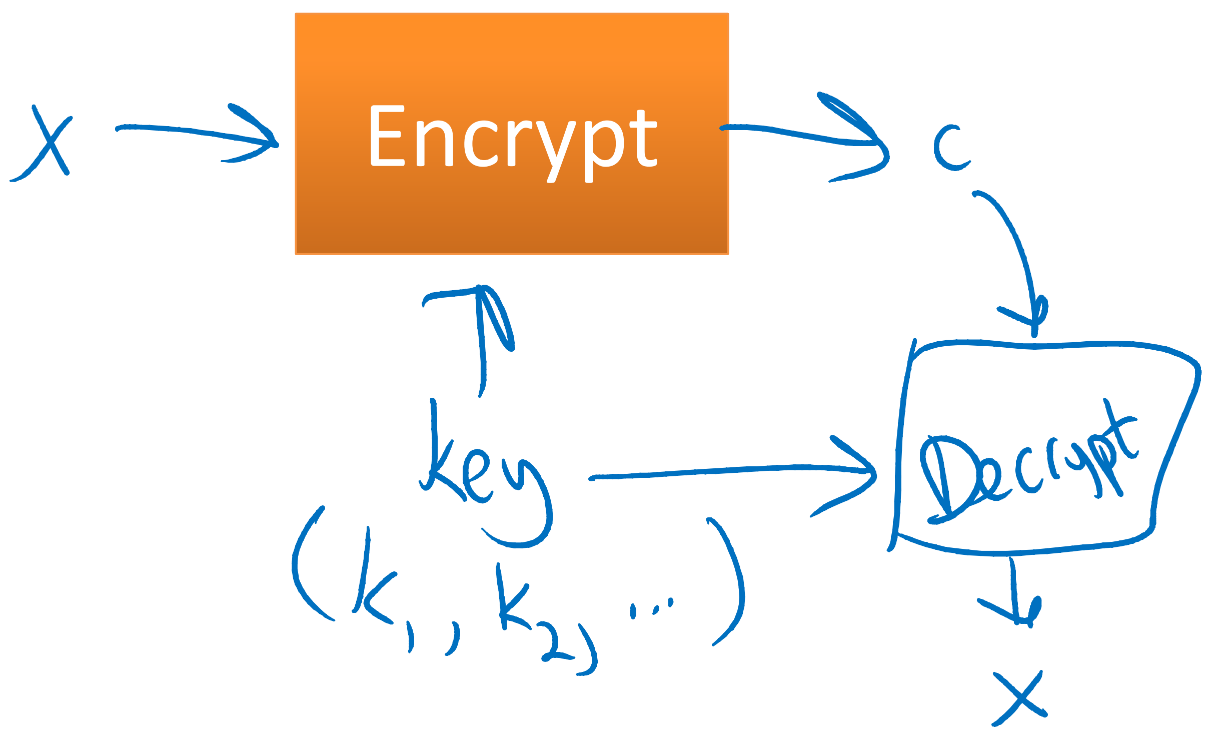
Computing with Meaningless Values?

Inputs		Output
a	b	x
a_0	b_0	x_0
a_0	b_1	x_0
a_1	b_0	x_0
a_1	b_1	x_1

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



Encryption



Logic with Privacy

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

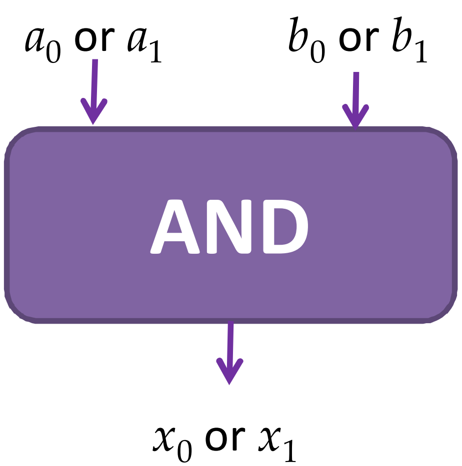
(i) $k = a_0, b_0$ → x_i → a_0 a_1

(ii) $k = b_0$

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!



Random Permutation

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: $a_0, a_1, b_0, b_1, x_0, x_1$

Garbled 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

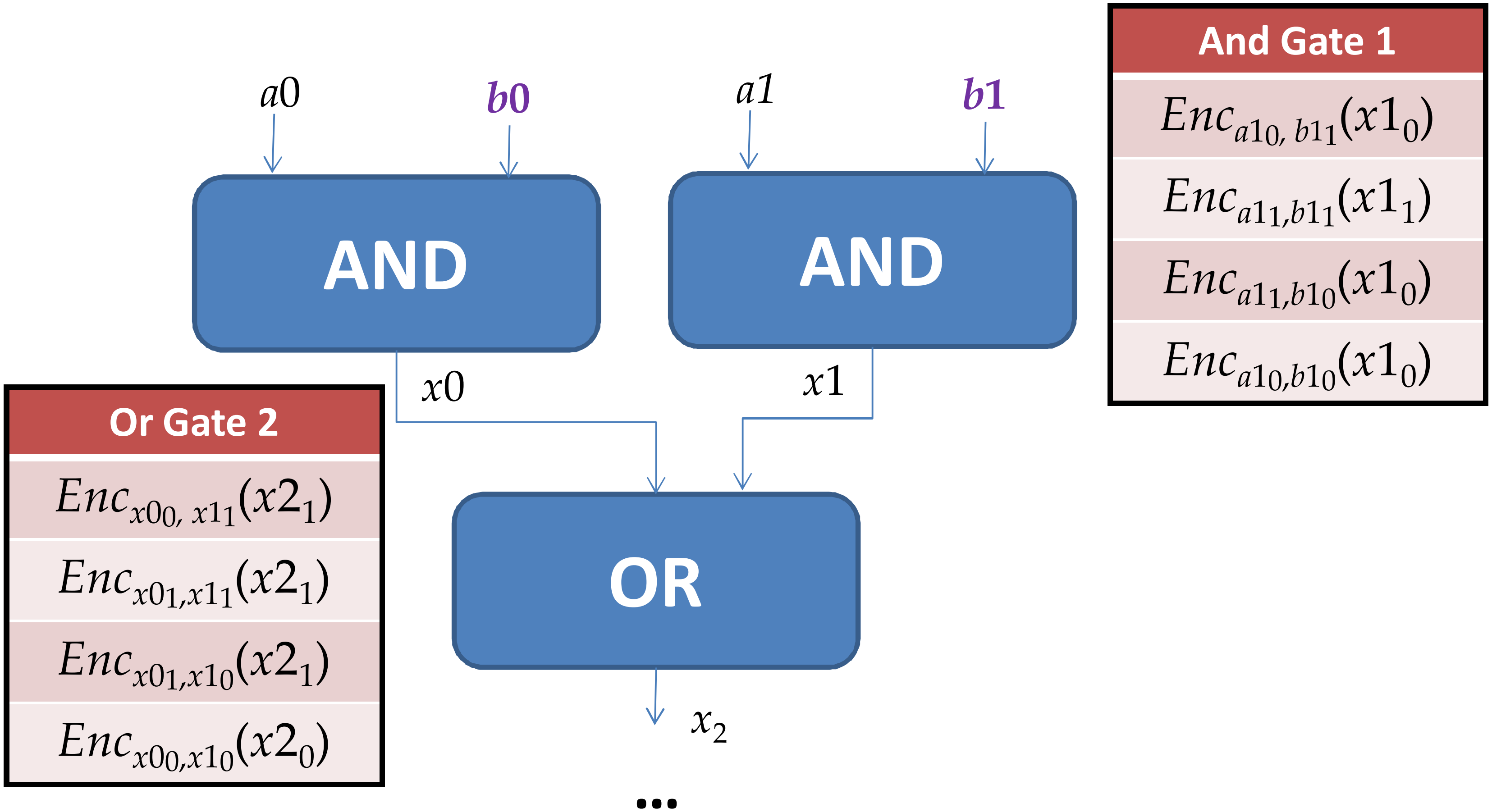
Learns nothing

Learns b_i (only)

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

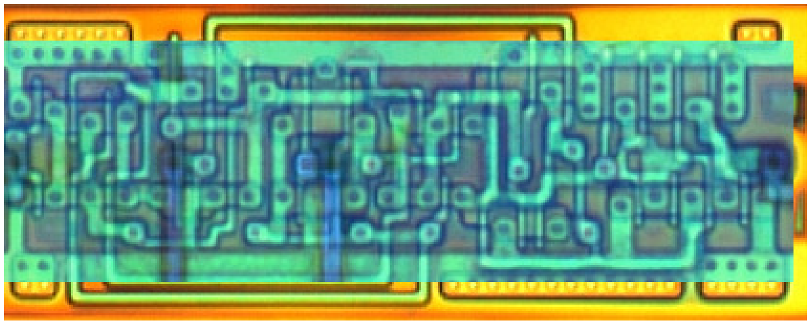
Rabin, 1981; Even, Goldreich, and Lempel, 1985; many subsequent papers

Chaining Garbled Circuits



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

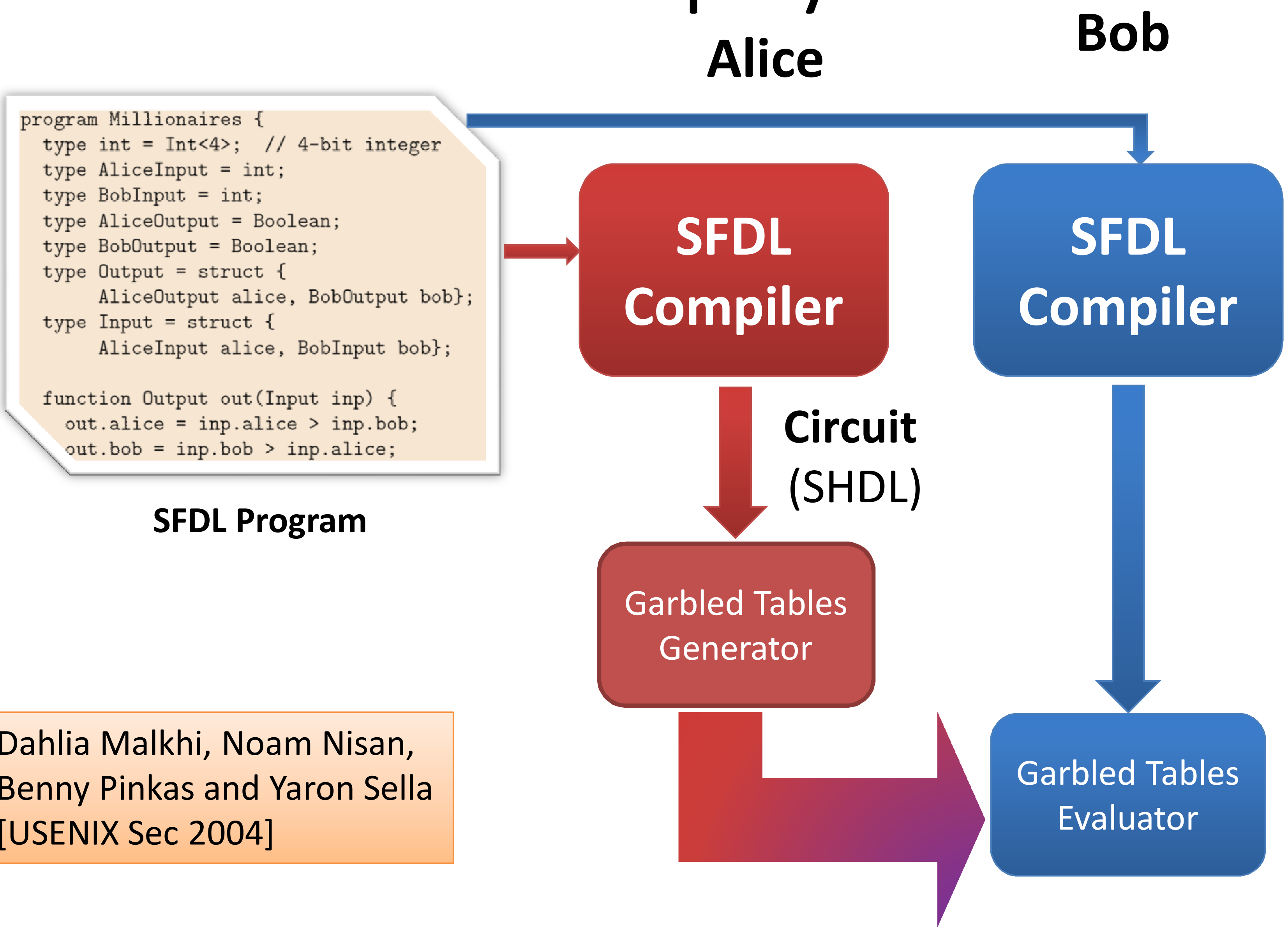
Building Computing Systems



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

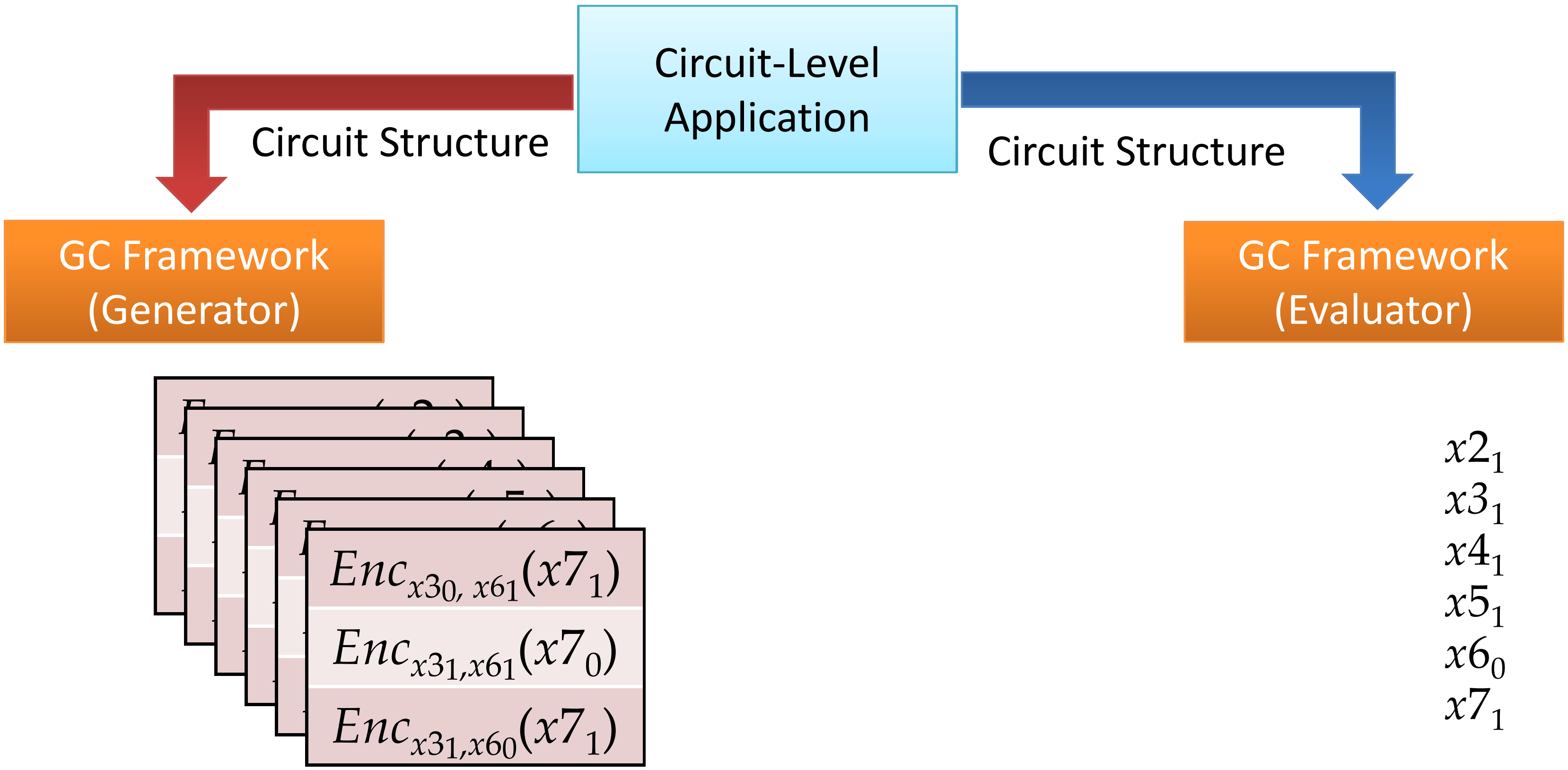
Digital Electronic Circuits	Garbled Circuits
Operate on **known data**	Operate on **encrypted wire labels**
One-bit logical operation requires moving a few electrons a few nanometers (hundreds of Billions per second)	One-bit logical operation requires performing (up to) 4 encryption operations: **very slow execution**
Reuse is great!	Reuse is not allowed for privacy: **huge circuits needed**

Fairplay



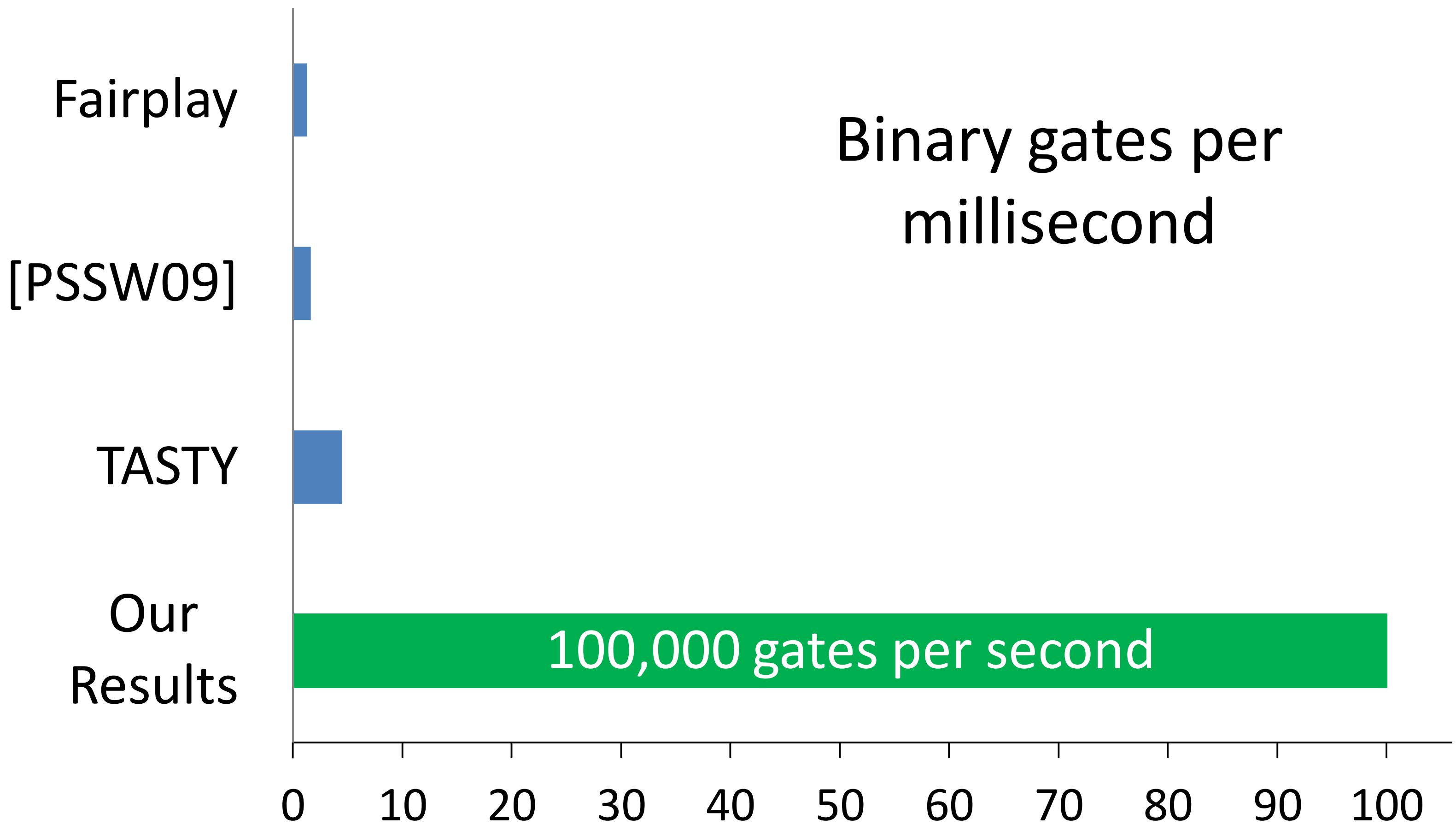
Dahlia Malkhi, Noam Nisan, Benny Pinkas and Yaron Sella [USENIX Sec 2004]

Faster Garbled Circuits

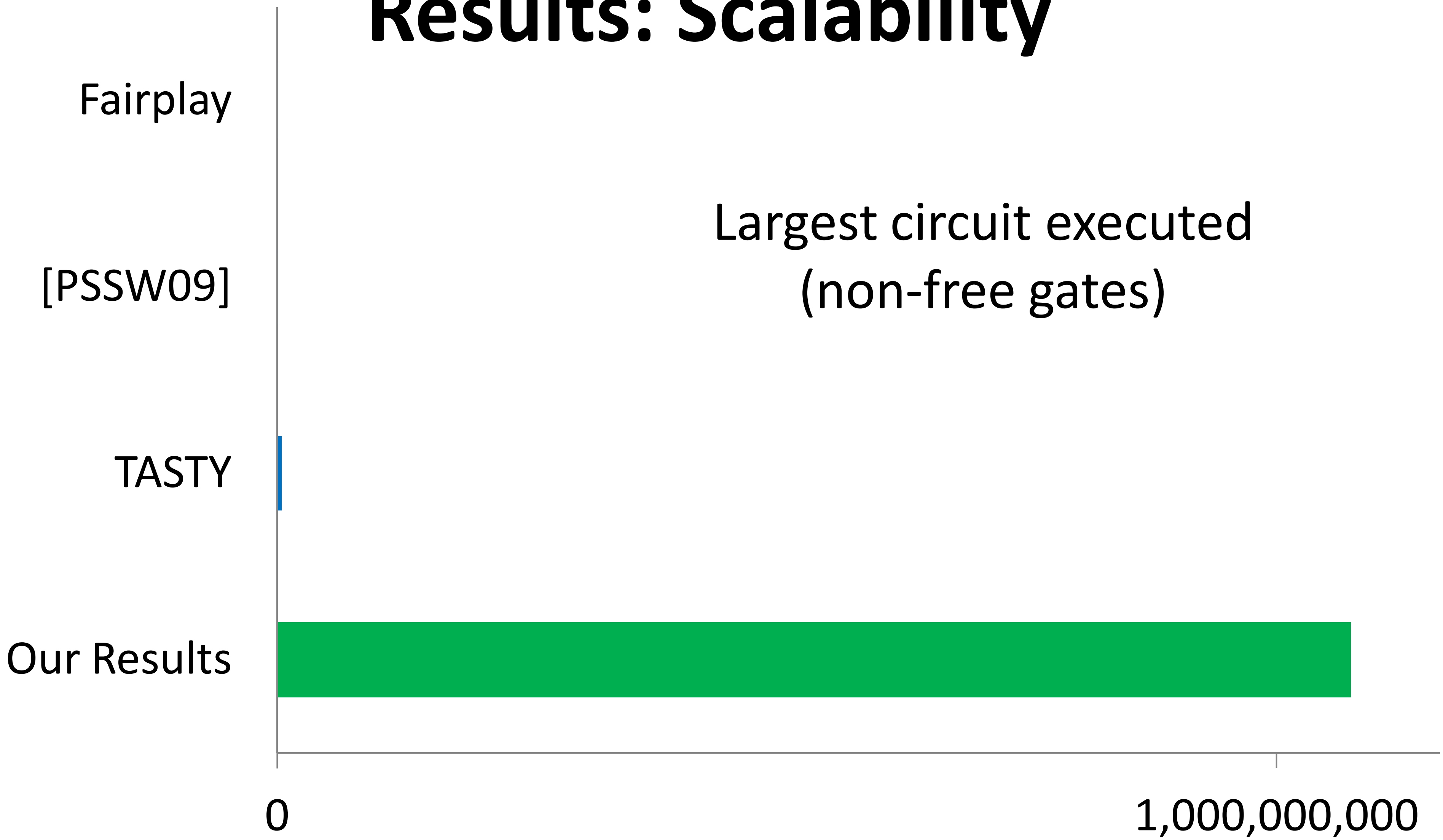


Gates can be evaluated as they are generated: **pipelining**
Gates can be evaluated in any topological sort order: **parallelizing**
Garbled evaluation can be **combined with normal execution**

Results: Performance



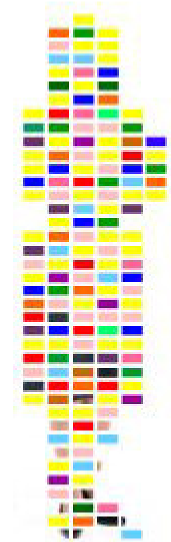
Results: Scalability



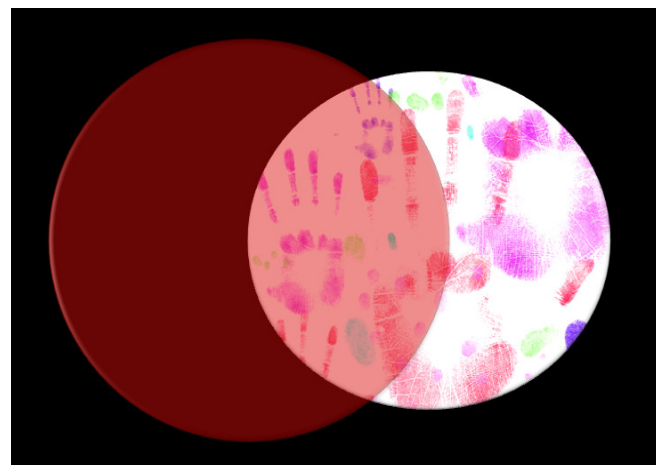
Applications

Privacy-Preserving
Biometric Matching

Private
Personal
Genomics



Private AES
Encryption



Private Set Intersection

Heterozygous Recessive Risk

Alice

Bob		**A**	**a**
	A	AA	Aa
	a	aA	aa

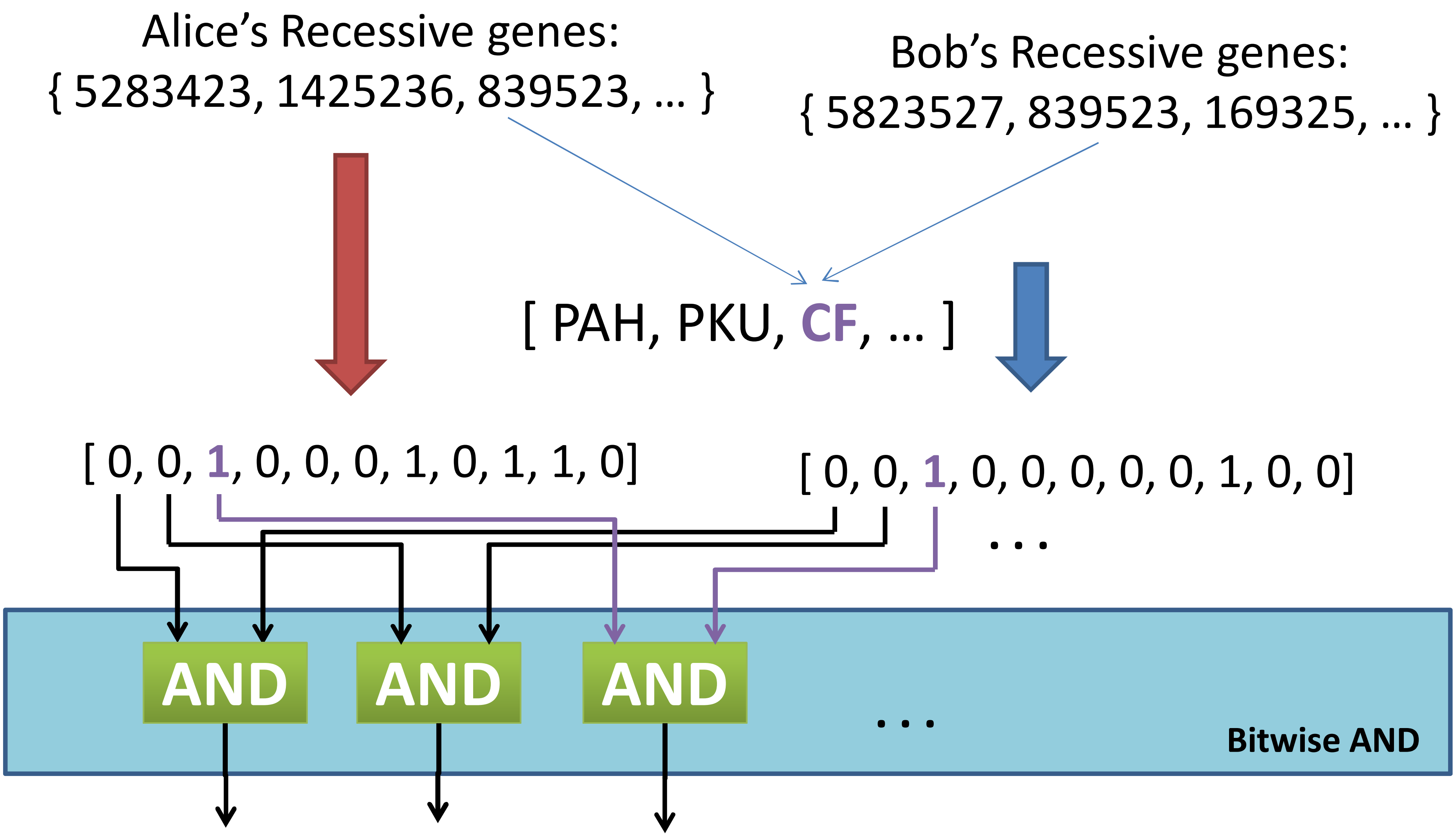
Aa ← carrier

aa ← cystic fibrosis

Alice's Heterozygous Recessive genes: { 5283423, 1425236, 839523, … }
Bob's Heterozygous Recessive genes: { 5823527, 839523, 169325, … }

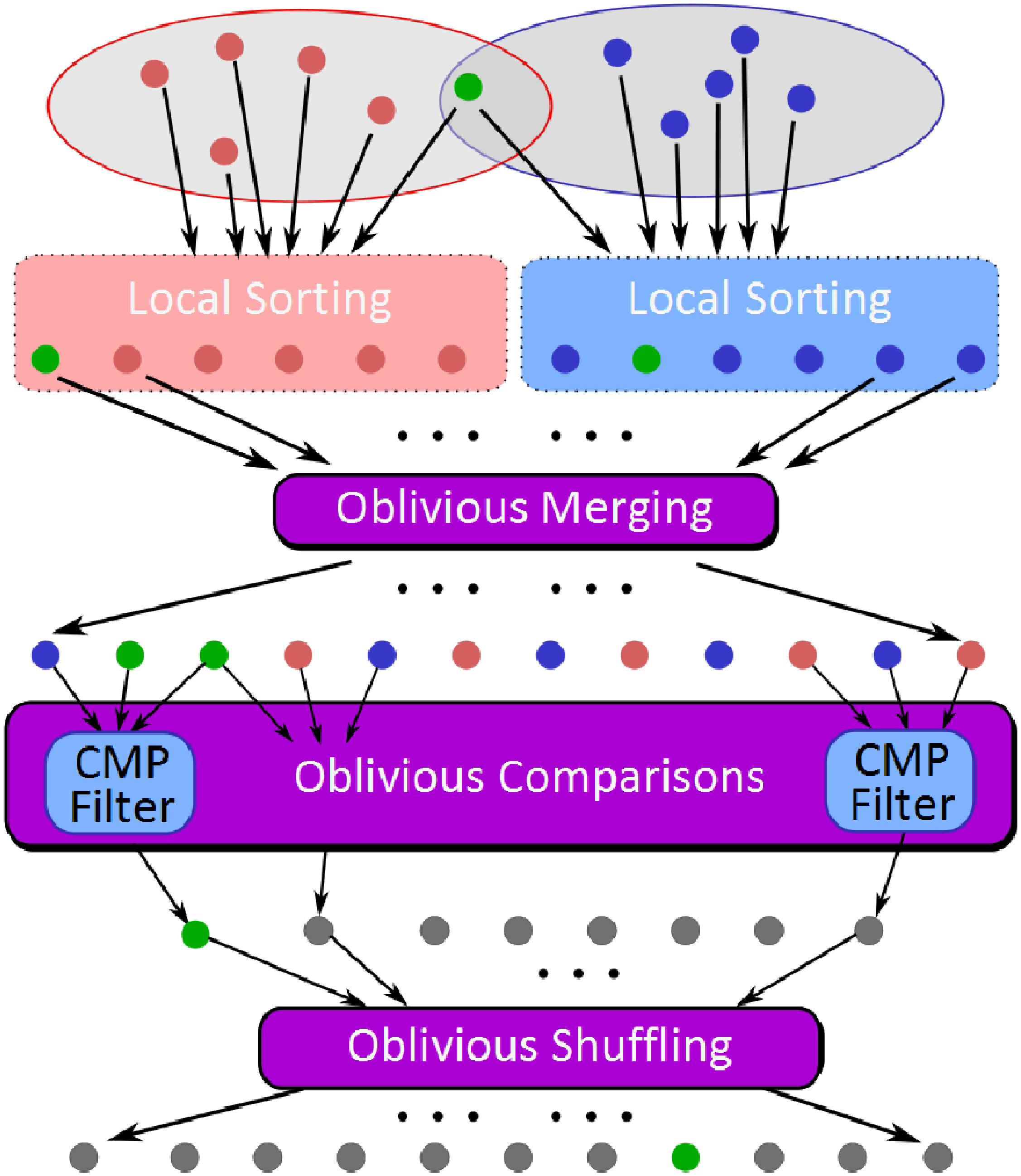
Goal: find the intersection of A and B

Bit Vector Intersection



Common Contacts

Sort-Compare-Shuffle

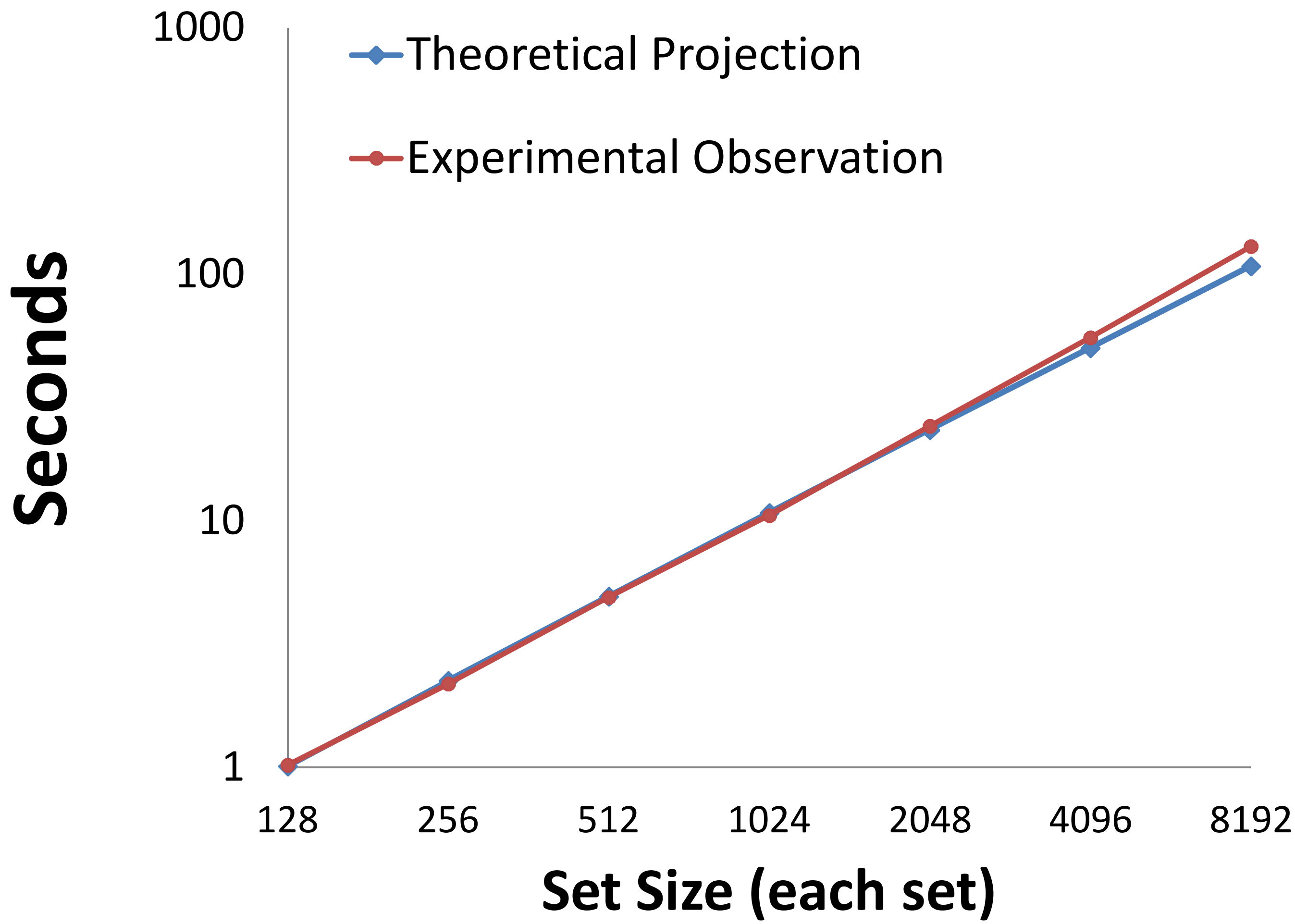


Sort: Take advantage of total order of elements

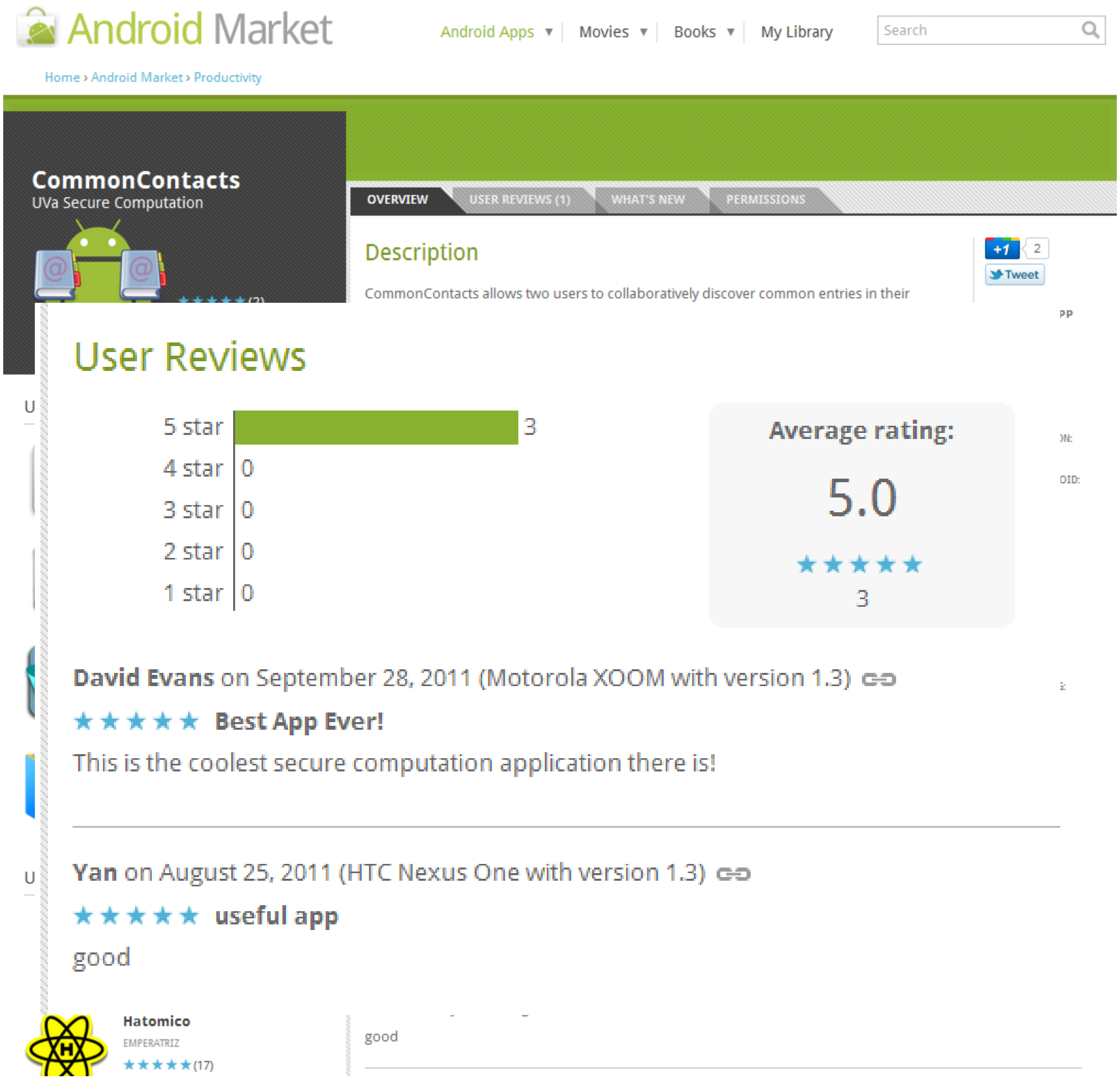
Compare adjacent elements

Shuffle to hide positions

SCS-WN Protocol Results

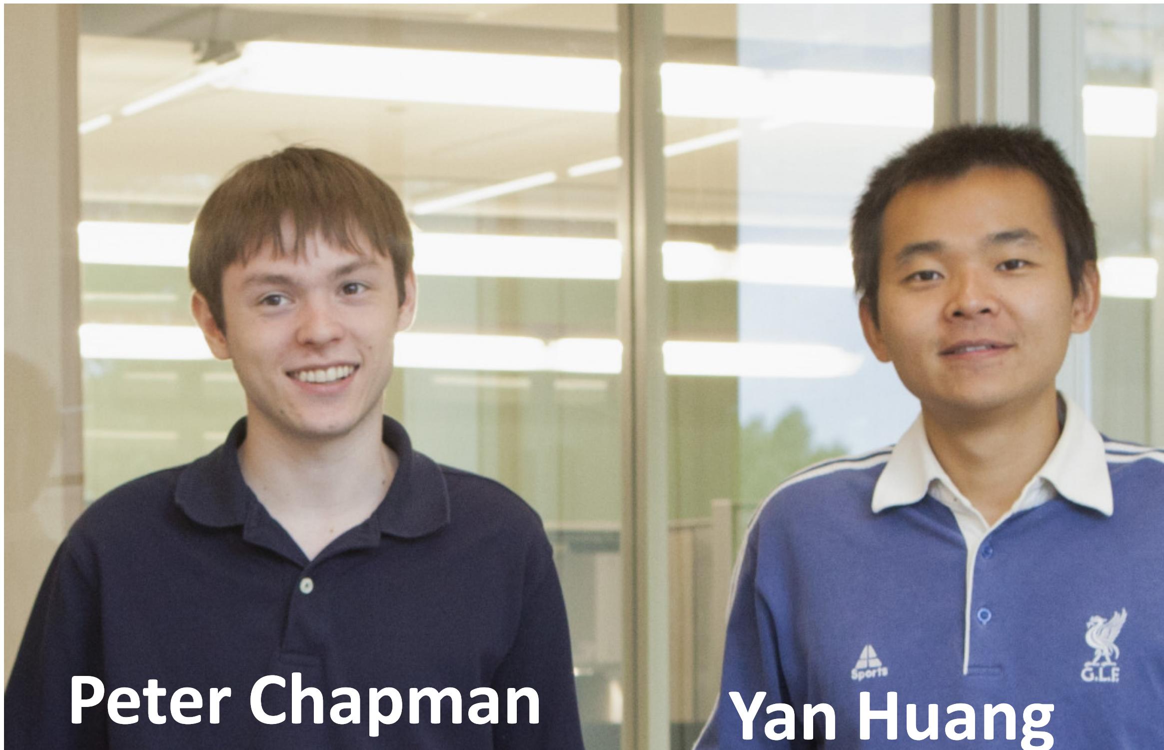


32-bit values


Android Market
Android Apps
Movies
Books
My Library
Search
Home › Android Market › Productivity
CommonContacts
UVa Secure Computation
OVERVIEW
USER REVIEWS (1)
WHAT'S NEW
PERMISSIONS
Description
+1 2
Tweet
CommonContacts allows two users to collaboratively discover common entries in their
User Reviews
5 star 3
4 star 0
3 star 0
2 star 0
1 star 0
Average rating:
5.0
3
David Evans on September 28, 2011 (Motorola XOOM with version 1.3)
Best App Ever!
This is the coolest secure computation application there is!
Yan on August 25, 2011 (HTC Nexus One with version 1.3)
useful app
good
Hatomico
EMPERATRIZ
(17)
good

	Problem	Best Previous Result	Our Result	Speedup
NDSS 2012	**Private Set Intersection** (contact matching, common disease carrier)	Competitive with best custom protocols, scales to millions of 32-bit elements		
USENIX Security 2011	**Hamming Distance** (Face Recognition)	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**

Research Group and Alumni



Peter Chapman
(UVa BACS 2012)

Yan Huang
(UVa Computer Science PhD Student)

Jonathan Katz
(University of Maryland)

Funding:
NSF, **MURI** (AFOSR), **Google**


MightBeEvil.com