NUS School of Computing

Computing Camp for JC Students
22-24 November 2010

What do human evolution, leukemia treatment, database design (and a few other things) have in common?

by
Prof. Wong Limsoon

Fun With Invariants

Limsoon Wong

NUS National University of Singapore

Plan

- **What is an invariant?**
  - Bet on color of the bean
  - 21 cards
- **Origin of Polynesians**
  - Problem solving by logical reasoning on invariants
- **Make a list sorted**
- **Design a good database**
  - Problem solving by rectifying violation of invariants
- **Diagnose leukemia**
  - Guilt by association of invariants
- **Make exponentiation faster**
  - Solution optimization by preserving invariants

What is an invariant?

NUS National University of Singapore

- **Suppose you have a bag of x red beans and y green beans**
- **Repeat the following:**
  - Remove 2 beans
  - If both green, discard both
  - If both red, discard one, put back one
  - If one green and one red, discard red, put back green
- **If one bean is left behind, can you predict its colour?**

Shall we bet on the color of the bean that is left behind?

Bet on the last green bean

- **Suppose you have a bag of x red beans and y green beans**
  - **When the parity of # of green beans (y) is odd, …**
- **Repeat the following:**
  - **Start with y=2n+1**
  - Remove 2 beans
  - If both green, discard both
    - **y=2n+1 → y=2n-1**
  - If both red, discard one, put back one
    - **y=2n+1 → y=2n+1**
  - If one green and one red, discard red, put back green
    - **y=2n+1 → y=2n+1**
- **If one bean is left behind, can you predict its colour?**
  - **y remains odd**
  - **⇒ Last bean must be green!**

Bet on the last red bean

- Suppose you have a bag of x red beans and y green beans
- Repeat the following:
  - Remove 2 beans
  - If both green, discard both
  - If both red, discard one, put back one
  - If one green and one red, discard red, put back green
- If one bean is left behind, can you predict its colour?

- When the parity of # of green beans (y) is even, …
- Start with y=2n
- y=2n → y=2n-2
- y=2n → y=2n
- y=2n → y=2n
- y remains even

⇒ Last bean must be red!

Bet on color of the last bean … and win!

- Suppose you have a bag of x red beans and y green beans
- Repeat the following:
  - Remove 2 beans
  - If both green, discard both
  - If both red, discard one, put back one
  - If one green and one red, discard red, put back green
- If one bean is left behind, can you predict its colour?

- If you start w/ odd # (even #) of green beans, there will always be an odd # (even #) of green beans in the bag

⇒ Parity of green beans is invariant

⇒ Bean left behind is green iff you start with odd # of green beans

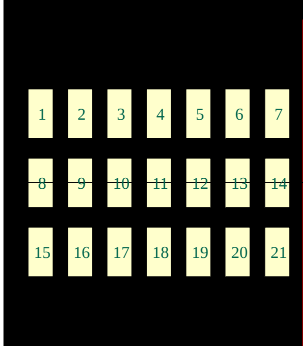
- **What have we just seen?**
- **Problem solving by logical reasoning on invariants**

Welcome to the Magical World…

The 21-Card Trick

1. Magician asks you to remember any one card from a deck of 21 cards as your card. Do not tell him what the card is
2. He deals the 21 cards face down, from top to bottom and left to right, into 3 equal piles
3. Next, he fans the piles to you and asks you to look for the pile of cards which contains your card and pass the pile back to him
4. Again, he stacks up the 3 piles on top of each other and redistribute, from top to bottom and left to right, into 3 equal piles
5. He repeats step (3) and (4) 2 more times
6. Finally, he deals your card right out from the rest of the 21 cards!

How does he manage that?!



13

The Trick

- The pile containing the card is being placed in the middle of the other 2 piles

- Imposing constraints on where the card can move to…

 This section of the ppt is courtesy of Toh Xiu Ping

14

The Invariant Underlying the Trick

Assuming the chosen card is in the first pile.

11	12	13	14	15	16	17
21	22	23	24	25	26	27
31	32	33	34	35	36	37

After the first distribution, …

21	24	27	13	16	32	35
22	25	11	14	17	33	36
23	26	12	15	31	34	37

After the second distribution, …

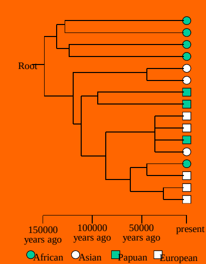
After the third distribution, …

 This section of the ppt is courtesy of Toh Xiu Ping

15

- What have we just seen?

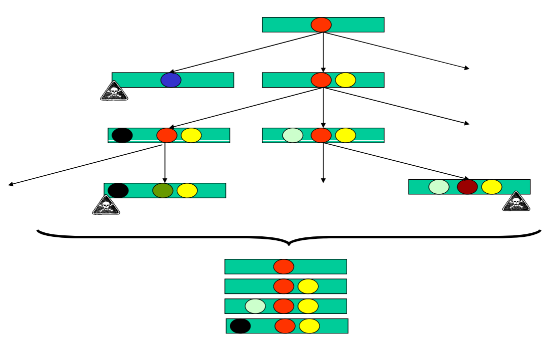
- Problem solving by logical reasoning on invariants (with a twist of injecting the invariant)



Where do Polynesians come from?

17

In the course of evolution…



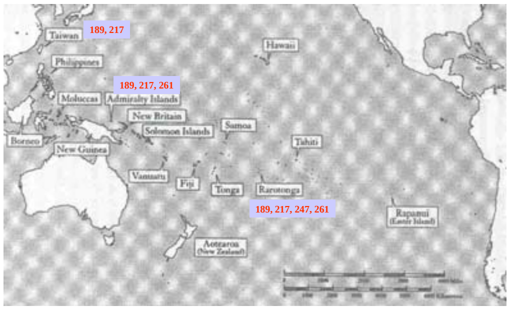
18

What is the invariant?

- Mitochrondrial DNA accumulates 1 mutation about every 10,000 years
- Human history is not so long relative to this

⇒ When a nucleotide in mitochrondrial DNA is mutated it stays mutated through future generations

Do Polynesians come from Asia or America?



Origin of Polynesians

- Common mitochondrial control seq from Rarotonga have variants at positions 189, 217, 247, 261. Less common ones have 189, 217, 261
- Seq from Taiwan natives have variants 189, 217
- Seq from regions in betw have variants 189, 217, 261.

- More 189, 217 closer to Taiwan. More 189, 217, 261 closer to Rarotonga
- 247 not found in America

⇒ Polynesians came from Taiwan!

- Taiwan seq sometimes have extra mutations not found in other parts

⇒ These are mutations that happened since Polynesians left Taiwan!

The "Invariant" Perspective

- The invariant:

  When a nucleotide in mitochrondrial DNA is mutated it stays mutated through future generations

- The lesson learned:

Figure out origins of Polynesians by logical reasoning on invariant

How to get a list sorted?

What makes a list a sorted list?

- What is a sorted list?

  A list L is sorted iff L[i] ≤ L[j] for all adjacent positions i < j

- So how do you make a list M become sorted?

```
While M[i] > M[j] for some adjacent positions i < j {
    swap M[i], M[j]
}
```


Sorting a list

- Invariant of sorted lists

  A list L is sorted iff L[i] ≤ L[j] for all adjacent positions i < j

- Making a list M become sorted:

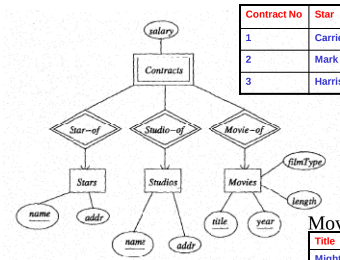
```
While M[i] > M[j] for some adjacent positions i < j {
    swap M[i], M[j]
}
```

- Find violation of the invariant
- Fix it
- When no more violation, the list must be sorted!

- **What have we just seen?**
- **Problem solving by rectifying violation of invariants**

What is a good database design?

Relational Data Model



Contracts

Contract No	Star	Studio	Title	Salary
1	Carrie Fisher	Fox	Star Wars	$$$
2	Mark Hamill	Fox	Star Wars	$$$
3	Harrison Ford	Fox	Star Wars	$$$

Stars

Name	Address
Carrie Fisher	Hollywood
Mark Hamill	Brentwood
Harrison Ford	Beverly Hills

Movies

Title	Year	Length	Film Type
Mighty Ducks	1991	104	Color
Wayne's World	1992	95	Color
Star Wars	1977	124	Color

Design Issues

- **How many possible alternate ways to represent movies using tables?**
- **Why this particular set of tables to represent movies?**
- **Indeed, why not use this alternative single table below to represent movies?**

Wrong Movies

Title	Year	Length	Film Type	Studio	Star
Star Wars	1997	124	Color	Fox	Carrie Fisher
Star Wars	1997	124	Color	Fox	Mark Hamill
Star Wars	1997	124	Color	Fox	Harrison Ford
Mighty Ducks	1991	104	Color	Disney	Emilio Estevez

Anomalies

- **What's wrong with the "Wrong Movies" table?**

Wrong Movies

Title	Year	Length	Film Type	Studio	Star
Star Wars	1997	124	Color	Fox	Carrie Fisher
Star Wars	1997	124	Color	Fox	Mark Hamill
Star Wars	1997	124	Color	Fox	Harrison Ford
Mighty Ducks	1991	104	Color	Disney	Emilio Estevez

- **Redundancy: Unnecessary repetition of info**
- **Update anomalies: If Star Wars is 125 min, we might carelessly update row 1 but not rows 2 & 3**
- **Deletion anomalies: If Emilio Estevez is deleted from stars of Mighty Ducks, we lose all info on that movie**

Functional Dependency

- **Functional dependency ($A_1, \ldots, A_n \rightarrow B_1, \ldots, B_m$)**
  - If two rows of a table R agree on attributes $A_1, \ldots, A_n$, then they must also agree on attributes $B_1, \ldots, B_m$
  - $\Rightarrow$ Values of B's depend on values of A's
- **Example: Title, Year → Length, Film Type, Studio**
- **FD ($A_1, \ldots, A_n \rightarrow B_1, \ldots, B_m$) is trivial if a B_i is an A_j**

Can you identify the FD's here?

Wrong Movies

Title	Year	Length	Film Type	Studio	Star
Star Wars	1997	124	Color	Fox	Carrie Fisher
Star Wars	1997	124	Color	Fox	Mark Hamill
Star Wars	1997	124	Color	Fox	Harrison Ford
Mighty Ducks	1991	104	Color	Disney	Emilio Estevez

- **Some FD's:**
  - Title, Year → Length
  - Title, Year → Film Type
  - Title, Year → Studio

Keys

- **Key**
  - A minimal set of attributes $\{A_1, \ldots, A_n\}$ that functionally determine all other attributes of a table
  - A key is trivial if it comprises the entire set of attributes of a table
- **Superkey**
  - A set of attributes that contains a key

Can you identify the superkeys here?

Wrong Movies

Title	Year	Length	Film Type	Studio	Star
Star Wars	1997	124	Color	Fox	Carrie Fisher
Star Wars	1997	124	Color	Fox	Mark Hamill
Star Wars	1997	124	Color	Fox	Harrison Ford
Mighty Ducks	1991	104	Color	Disney	Emilio Estevez

- **Superkeys :**
  - Any set of attributes that contains {Title, Year, Star} as a subset

Boyce-Codd Normal Form

- **A relation R is in Boyce-Codd Normal Form iff whenever there is a nontrivial FD $(A_1, \ldots, A_n \rightarrow B_1, \ldots, B_m)$ for R, it is the case that $\{A_1, \ldots, A_n\}$ is a superkey for R**
- Theorem A1 (Codd, 1972)

  A database design has no anomalies due to FD iff all its relations are in Boyce-Codd Normal Form

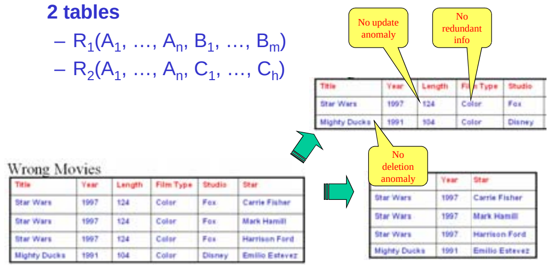
How is BCNF violated here?

Title	Year	Length	Film Type	Studio	Star
Star Wars	1997	124	Color	Fox	Carrie Fisher
Star Wars	1997	124	Color	Fox	Mark Hamill
Star Wars	1997	124	Color	Fox	Harrison Ford
Mighty Ducks	1991	104	Color	Disney	Emilio Estevez

- **A nontrivial FD:**
  - Title, Year → Length, Film Type, Studio
  - The LHS not superset of the key {Title,Year, Star}
  - ⇒ Violate BCNF!
- **Anomalies are due to FD's whose LHS is not superkey**

Towards a Better Design

- **Use an offending FD $(A_1, \ldots, A_n \rightarrow B_1, \ldots, B_m)$ to decompose $R(A_1, \ldots, A_n, B_1, \ldots, B_m, C_1, \ldots, C_h)$ into 2 tables**
  - $R_1(A_1, \ldots, A_n, B_1, \ldots, B_m)$
  - $R_2(A_1, \ldots, A_n, C_1, \ldots, C_h)$



Title	Year	Length	Film Type	Studio	Star
Star Wars	1997	124	Color	Fox	Carrie Fisher
Star Wars	1997	124	Color	Fox	Mark Hamill
Star Wars	1997	124	Color	Fox	Harrison Ford
Mighty Ducks	1991	104	Color	Disney	Emilio Estevez

Title	Year	Length	Film Type	Studio
Star Wars	1997	124	Color	Fox
Mighty Ducks	1991	104	Color	Disney

	Year	Star
Star Wars	1997	Carrie Fisher
Star Wars	1997	Mark Hamill
Star Wars	1997	Harrison Ford
Mighty Ducks	1991	Emilio Estevez

The "Invariant" Perspective

- **The invariants:**

  BCNF is an invariant of a good database design

- **The lesson learned:**

Deliver a better database design by fixing violated invariants

Impact

ORACLE CORPORATION

Q3 FISCAL 2010 FINANCIAL RESULTS
CONDENSED CONSOLIDATED STATEMENTS OF OPERATIONS
($ in millions, except per share data)

	Three Months Ended February 28,				% Increase (Decrease) in US $
	2010	% of Revenues	2009	% of Revenues	
REVENUES					
New software licenses	$ 1,718	27%	$ 1,516	28%	13%
Software license updates and product support	3,297	51%	2,917	53%	13%
Software Revenues	5,015	78%	4,433	81%	13%
Hardware systems products	273	4%	-	0%	*
Hardware systems support	185	3%	-	0%	*
Hardware Systems Revenues	458	7%	-	0%	*
Services	931	15%	1,020	19%	(9%)
Total Revenues	6,404	100%	5,453	100%	17%

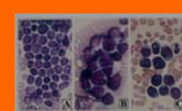
Impact

- **Data from US**

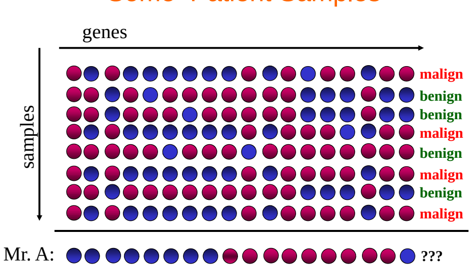
Occupational Title	SOC Code	Employment, 2008	Projected Employment, 2018	Change, 2008-18 Number	Percent
Computer network, systems, and database administrators	—	961,200	1,247,800	286,600	30
Database administrators	15-1061	120,400	144,700	24,400	20
Network and computer systems administrators	15-1071	339,500	418,400	78,900	23
Network systems and data communications analysts	15-1081	292,000	447,800	155,800	53
All other computer specialists	15-1099	209,300	236,800	27,500	13

- **Data from Singapore**
  - 3% of unfilled ICT vacancies are in db mgmt

Diagnosing Leukemias

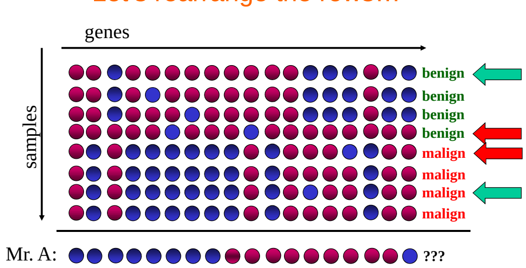


Some Patient Samples



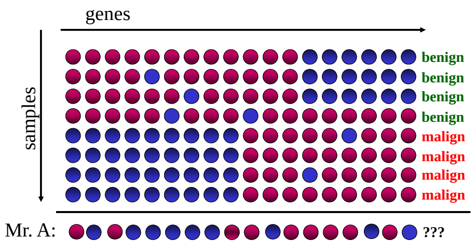
- **Does Mr. A have cancer?**

Let's rearrange the rows…

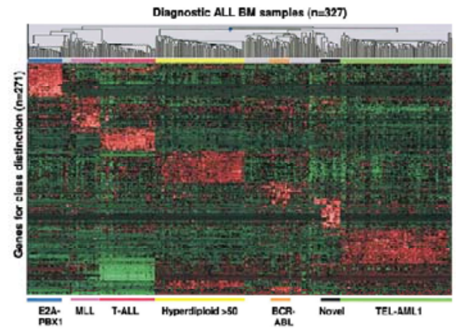


- **Does Mr. A have cancer?**

and the columns too…



Invariant Profile of Leukemia Subtypes



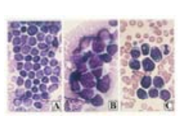
- **What have we just seen?**
- **Guilt by association of invariants**

Making an Impact:
Leukemia Diagnosis Revisited

Childhood Acute Lymphoblastic Leukemia

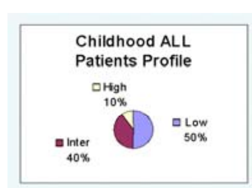
- **Major subtypes: T-ALL, E2A-PBX, TEL-AML, BCR-ABL, MLL genome rearrangements, Hyperdiploid>50**
- **Diff subtypes respond differently to same Tx**
- **Over-intensive Tx**
  - Development of secondary cancers
  - Reduction of IQ
- **Under-intensiveTx**
  - Relapse

- **The subtypes look similar**



- **Conventional diagnosis**
  - Immunophenotyping
  - Cytogenetics
  - Molecular diagnostics
  - ⇒ Unavailable in developing countries

Patient Profiles & Treatment Costs



- **2000 new cases a year in ASEAN countries**

- **Treatment for childhood ALL over 2 yrs**
  - Intermediate intensity: US$60k
  - Low intensity: US$36k
  - High intensity: US$72k
- **Treatment for relapse: US$150k**
- **Cost for side-effects: Unquantified**

Why not high/low intensity to everyone?

- **High-intensity Tx**
  - Over intensive for 90% of patients, thus a lot more side effects
  - US$144m (US$72k * 2000) for high-intensity tx

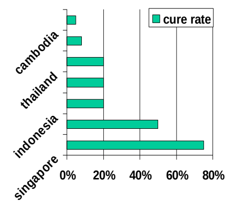
⇒ **Total US$144m/yr plus un-quantified costs for dealing with side effects**

- **Low-intensity Tx**
  - Under intensive for 50% of patients, thus a lot more relapse
  - US$72m (US$36k * 2000) for low-intensity tx
  - US$150m (US$150k * 2000 * 50%) for relapse tx

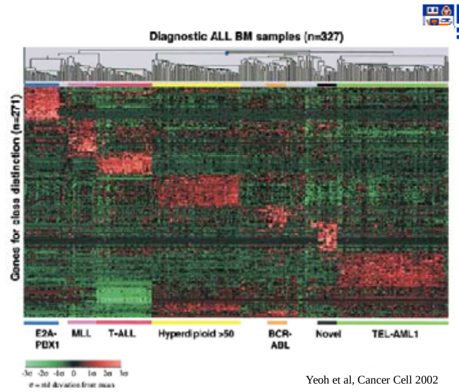
⇒ **Total US$222m/yr**

Current Situation

- **Intermediate intensity conventionally applied in ASEAN countries**



- **Over intensive for 50% of patients, thus more side effects**
- **Under intensive for 10% of patients, thus more relapse**
- **US$120m (US$60k * 2000) for intermediate intensity tx**
- **US$30m (US$150k * 2000 * 10%) for relapse tx**
- **Total US$150m/yr plus un-quantified costs for dealing with side effects**



Yeoh et al, Cancer Cell 2002

Exploit Invariant Gene Expr Profiles

- **Low intensity applied to 50% of patients**
- **Intermediate intensity to 40% of patients**
- **High intensity to 10% of patients**

⇒ **Reduced side effects**
⇒ **Reduced relapse**
⇒ **75-80% cure rates**

- **US$36m (US$36k * 2000 * 50%) for low intensity**
- **US$48m (US$60k * 2000 * 40%) for intermediate intensity**
- **US$14.4m (US$72k * 2000 * 10%) for high intensity**
- **Total US$98.4m/yr**

⇒ **Save US$51.6m/yr**

Yeoh et al, Cancer Cell 2002

How to take exponentiation faster?

- **What does this program do?**

F(a, 0) = 1
F(a, n+1) = a * F(a, n)

- **We see that**

F(a, n) = a^n

Exponentiation

$$a^n = \underbrace{a \times \cdots \times a}_{n},$$

Playing the invariant…

- What does this program do?

  F(a, 0) = 1
  F(a, n+1) = a * F(a, n)

- We see that ← defn

  F(a, n) = a^n ← invariant

- Then
- F(a, 2*n) = a^{2*n} ← by inv
  = a^n * a^n ← by inv
  = y * y where y = F(a, n)
- F(a, 2*n+1) = a^{2*n+1} ← by inv
  = a * a^n * a^n
  = a * y * y where F(a, n) ← by inv
- So we get …

What's the difference?

- Original program:

  F(a, 0) = 1
  F(a, n+1) = a * F(a, n)

- Cost of F(a, n) = n

- New program:

  F(a, 0) = 1
  F(a, 1) = a
  F(a, n) = if n is odd
  then a * y * y
  else y * y
  where y = F(a, n div 2)

  (Parity can be tested by checking least significant bit)
  (Div2 can be implemented by bit shifting)

- Cost of F(a, n) = $\log_2 n$

exponentially Faster!

n	log n	call sequence		
8	3	4	2	1
9	3	4	2	1
10	3	5	2	1
11	3	5	2	1

- **What have we just seen?**
- **Optimizing a solution by preserving invariant**

How to make computers safer?

COMPUTERWORLD An IDG company

RSA: Microsoft on 'rootkits': Be afraid, be very afraid
Rootkits are a new generation of powerful system-monitoring programs

News Story by Paul Roberts

FEBRUARY 17, 2005 (IDG NEWS SERVICE) - Microsoft Corp. security researchers are warning about a new generation of powerful system-monitoring programs, or "rootkits," that are almost impossible to detect using current security products and could pose a serious risk to corporations and individuals……the only reliable way to remove kernel rootkits is to completely erase an infected hard drive and reinstall the operating system from scratch……

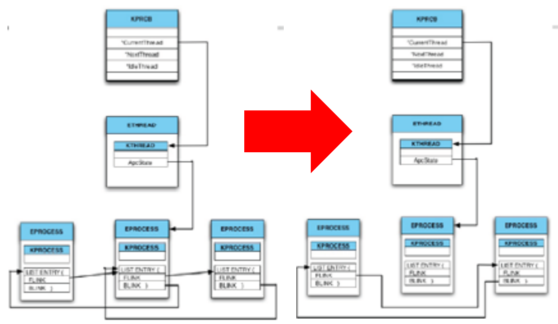
Credit: Bill Arbaugh

Rootkit Problem

- **Traditional rootkits**
  - Modify scalar invariants in OS
    - kernel text
    - interrupt table
    - syscall table
- **Modern rootkits**
  - Direct Kernel Object Manipulation (DKOM)
  - Rather than modify scalar invariants in OS, data of kernel are modified to:
    - Hide processes
    - Increase privilege level

Credit: Bill Arbaugh

Hiding a window process



Semantic integrity

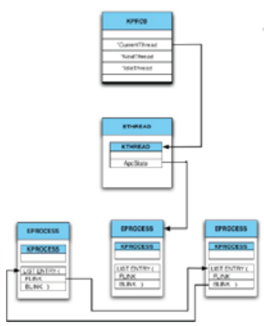
- **Current integrity monitoring systems focus on the scalar nature of the monitored data**
  – Work for scalar (i.e., invariant) data
  – Don't work for non-scalar data
- **Semantic integrity**
  – Monitor non-invariant portions of a system via predicates that remain valid during the proper operation of the system
  – I.e., monitor invariant dynamic properties!

Credit: Bill Arbaugh

DKOM Example

- **Semantic integrity predicate (ie., dynamic invariant) is**
- **There is no thread such that its parent process is not on the process list**

⇒kHIVE (contains 20k other predicates)



- **What have we just seen?**
- **Maintain computer safety by checking violation of invariants!**

Impact

- **2008: Komoku (kHIVE) acquired by Microsoft**
- **2009: Put into MS Security Essentials (~4m hosts)**
- **2010: Put into Windows Update (~500m hosts)**

"There is no other field out there where you can get right out of university and define substantial aspects of a product that is going to go out and over 100 million people are going to use it". ---Bill Gate

Remarks

What have we learned?

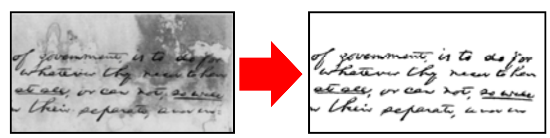
- **Invariant is a fundamental property of many problems**
- **Paradigms of problem solving**
  – Problem solving by logical reasoning on invariants
  – Problem solving by rectifying/monitoring violation of invariants
  – Guilt by association of invariants
  – Solution optimization by preserving invariants

I didn't get to telling you yet, but …

- **Every time you write a loop in a program, it involves an invariant**
- **Every time you do a recursive function call, it involves an invariant**
- **Every time you do an induction proof, it involves an invariant**
- **… Computing is about discovering, understanding, exploiting, and having fun with invariants!**

A Test: "Restoring" Historic Documents

Suggest a way to digitally restore damaged historic documents



Enjoy!