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

by Prof. Wong Limsoon (黃任祥)

Plan

- What is an invariant?
  - Bet on color of the bean
  - 21 cards
- Origin of Polynesians
- Make a list sorted
- Design a good database
- Diagnose leukemia
- Make exponentiation faster
- Problem solving by logical reasoning on invariants (用不變量的邏輯推理解決問題)
- Problem solving by rectifying violation of invariants (糾正被違反的不變量解決問題)
- Guilt by association of invariants (近墨者黑)
- Solution optimization by preserving invariants (用不變量優化解決方案)
What is an invariant?

**中文**
何为不变量

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

- 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?

**Bet on the last green 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?

**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?
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

\[ \Rightarrow \text{Parity of green beans is invariant} \]

\[ \Rightarrow \text{Bean left behind is green iff you start with odd # of green beans} \]

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?!
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...

The Invariant Underlying the Trick

Assuming the chosen card is in the first pile.

<table>
<thead>
<tr>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
</tr>
</tbody>
</table>

After the first distribution, ...

<table>
<thead>
<tr>
<th>21</th>
<th>24</th>
<th>27</th>
<th>13</th>
<th>16</th>
<th>32</th>
<th>35</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>25</td>
<td>11</td>
<td>14</td>
<td>17</td>
<td>33</td>
<td>36</td>
</tr>
<tr>
<td>23</td>
<td>26</td>
<td>12</td>
<td>15</td>
<td>31</td>
<td>34</td>
<td>37</td>
</tr>
</tbody>
</table>

After the second distribution, ...

After the third distribution, ...

What have we just seen?

- Problem solving by logical reasoning on invariants (用不変量的邏輯推理解決問題)

Where do Polynesians come from?
In the course of evolution...

What is the invariant?

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

⇒ When a nucleotide in mitochondrial 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
- More 189, 217 closer to Taiwan. More 189, 217, 261 closer to Rarotonga
- 247 not found in America
  ⇒ Polynesians came from Taiwan!
- Seq from Taiwan natives have variants 189, 217
- Seq from regions in betw have variants 189, 217, 261.
- 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 mitochondrial 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?

• 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 makes a list a sorted list? (何為列表排序)

• 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]
  }
What have we just seen?

Problem solving by rectifying violation of invariants (糾正被違反的不變量解決問題)

What is a good database design?

Relational Data Model

- Data are represented as a two-dimensional table
- It is a logical representation, not a physical representation
  - Ordering of the rows is irrelevant
  - Ordering of the columns is irrelevant
  - How the rows and columns of a table are stored is irrelevant
  - …

Example

<table>
<thead>
<tr>
<th>Contract No</th>
<th>Star</th>
<th>Studio</th>
<th>Title</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Carrie Fisher</td>
<td>Fox</td>
<td>Star Wars</td>
<td>$$$</td>
</tr>
<tr>
<td>2</td>
<td>Mark Hamill</td>
<td>Fox</td>
<td>Star Wars</td>
<td>$$$</td>
</tr>
<tr>
<td>3</td>
<td>Harrison Ford</td>
<td>Fox</td>
<td>Star Wars</td>
<td>$$$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Title</th>
<th>Year</th>
<th>Length</th>
<th>Film Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Might Ducks</td>
<td>1996</td>
<td>88</td>
<td>Color</td>
</tr>
<tr>
<td>Wayne’s World</td>
<td>1992</td>
<td>95</td>
<td>Color</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1977</td>
<td>124</td>
<td>Color</td>
</tr>
</tbody>
</table>

Stars

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carrie Fisher</td>
<td>Hollywood</td>
</tr>
<tr>
<td>Mark Hamill</td>
<td>Brentwood</td>
</tr>
<tr>
<td>Harrison Ford</td>
<td>Beverly Hills</td>
</tr>
</tbody>
</table>

Movies
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?

<table>
<thead>
<tr>
<th>Title</th>
<th>Year</th>
<th>Length</th>
<th>Film Type</th>
<th>Studio</th>
<th>Star</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Carrie Fisher</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Mark Hamill</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Harrison Ford</td>
</tr>
<tr>
<td>Mighty Ducks</td>
<td>1991</td>
<td>104</td>
<td>Color</td>
<td>Disney</td>
<td>Emilio Estevez</td>
</tr>
</tbody>
</table>

Wrong Movies

Anomalies

- What’s wrong with the “Wrong Movies” table?

<table>
<thead>
<tr>
<th>Title</th>
<th>Year</th>
<th>Length</th>
<th>Film Type</th>
<th>Studio</th>
<th>Star</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Carrie Fisher</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Mark Hamill</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Harrison Ford</td>
</tr>
<tr>
<td>Mighty Ducks</td>
<td>1991</td>
<td>104</td>
<td>Color</td>
<td>Disney</td>
<td>Emilio Estevez</td>
</tr>
</tbody>
</table>

- 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 \(\rightarrow\) 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?

<table>
<thead>
<tr>
<th>Title</th>
<th>Year</th>
<th>Length</th>
<th>Film Type</th>
<th>Studio</th>
<th>Star</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Carrie Fisher</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Mark Hamill</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Harrison Ford</td>
</tr>
<tr>
<td>Mighty Ducks</td>
<td>1991</td>
<td>104</td>
<td>Color</td>
<td>Disney</td>
<td>Emilio Estevez</td>
</tr>
</tbody>
</table>

- Some FD’s:
  - Title, Year \(\rightarrow\) Length
  - Title, Year \(\rightarrow\) Film Type
  - Title, Year \(\rightarrow\) 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?

<table>
<thead>
<tr>
<th>Title</th>
<th>Year</th>
<th>Length</th>
<th>Film Type</th>
<th>Studio</th>
<th>Star</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Carrie Fisher</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Mark Hamill</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Harrison Ford</td>
</tr>
<tr>
<td>Mighty Ducks</td>
<td>1991</td>
<td>104</td>
<td>Color</td>
<td>Disney</td>
<td>Emilio Estevez</td>
</tr>
</tbody>
</table>

- **Superkeys**:
  - Any set of attributes that contains \( \{\text{Title}, \text{Year}, \text{Star}\} \) 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?

<table>
<thead>
<tr>
<th>Title</th>
<th>Year</th>
<th>Length</th>
<th>Film Type</th>
<th>Studio</th>
<th>Star</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Carrie Fisher</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Mark Hamill</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1997</td>
<td>124</td>
<td>Color</td>
<td>Fox</td>
<td>Harrison Ford</td>
</tr>
<tr>
<td>Mighty Ducks</td>
<td>1991</td>
<td>104</td>
<td>Color</td>
<td>Disney</td>
<td>Emilio Estevez</td>
</tr>
</tbody>
</table>

- **A nontrivial FD**:
  - Title, Year \( \rightarrow \) Length, Film Type, Studio
  - The LHS is not superset of the key \( \{\text{Title}, \text{Year}, \text{Star}\} \)
  \( \Rightarrow \) Violate BCNF!

- Anomalies are due to FD’s whose LHS is not superkey
Towards a Better Design

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

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

Diagnosing Leukemias

- Some Patient Samples
  - Mr. A: Malign, benign, malign, benign, malign, benign, malign, benign, malign

- 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
  (近墨者黑)
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-intensive Tx
    - Relapse
- Conventional diagnosis
  - Immunophenotyping
  - Cytogenetics
  - Molecular diagnostics
  - unavailable in developing countries

Patient Profiles & Treatment Costs

- 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
- 2000 new cases a year in ASEAN countries

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
- 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$144m/yr plus unquantified costs for dealing with side effects
⇒ 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 * 10%) for relapse tx
- Total US$150m/yr plus unquantified costs for dealing with side effects

Exploit Invariant Gene Expr Profiles

- Low intensity applied to 50% of patients
- Intermediate intensity to 40% of patients
- High intensity to 10% of patients
- 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
- Reduced side effects
- Reduced relapse
- 75-80% cure rates
- Total US$98.4m/yr
- Save US$51.6m/yr

How to take exponentiation faster?
Exponentiation

What does this program do?

\[ F(a, 0) = 1 \]
\[ F(a, n+1) = a \cdot F(a, n) \]

We see that

\[ F(a, n) = a^n \]

Then

\[ F(a, 2n) = a^{2n} = (a^n)^2 \]
\[ = y \cdot y \text{ where } y = F(a, n) \]

So we get ...

What's the difference?

Original program:

\[ F(a, 0) = 1 \]
\[ F(a, n+1) = a \cdot F(a, n) \]

Cost of \( F(a, n) = n \)

New program:

\[ F(a, 0) = 1 \]
\[ F(a, 1) = a \]
\[ F(a, n) = \text{if } n \text{ is odd then } a \cdot y \cdot y \text{ else } y \cdot y \text{ where } y = F(a, n \div 2) \]

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

What have we just seen?

Optimizing a solution by preserving invariant (用不變量優化解決方案)
How to make computers safer?

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

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 the infected hard drive and reinstall the operating system from scratch....
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!

DKOM Example

- Semantic integrity predicate (i.e., dynamic invariant) is

  =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!