Ẩn tin mật trên văn bản (phần 2)

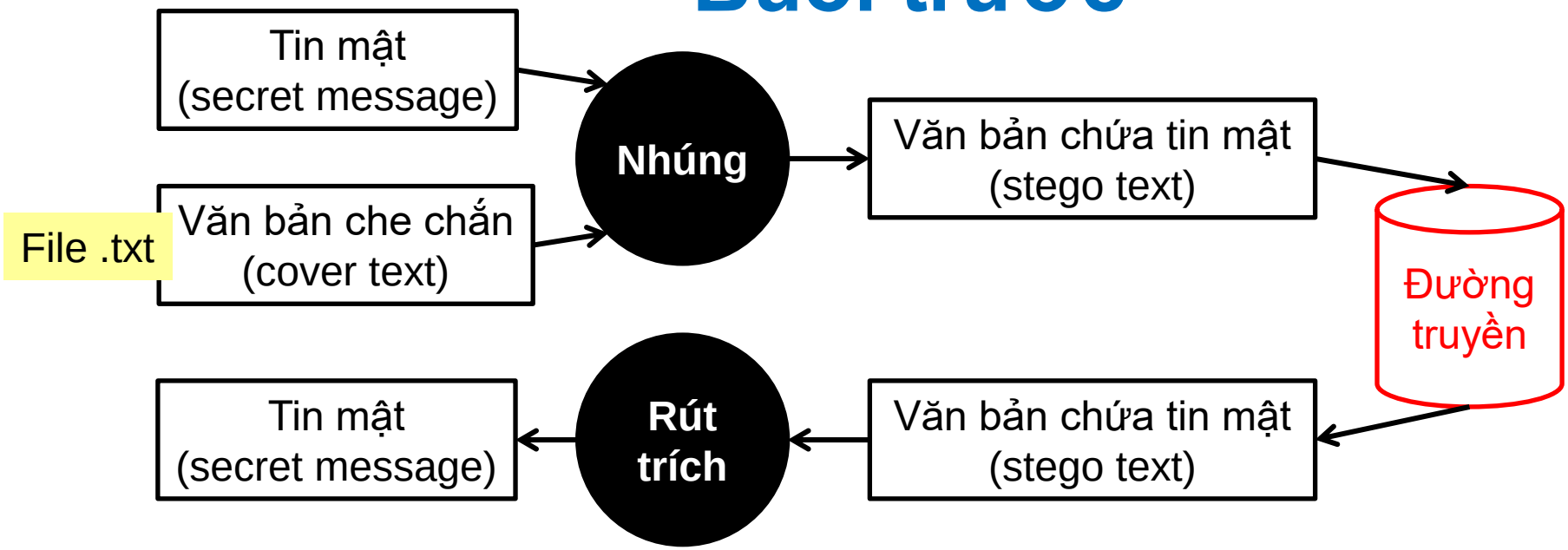
KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Thay đổi về cách đánh giá môn học

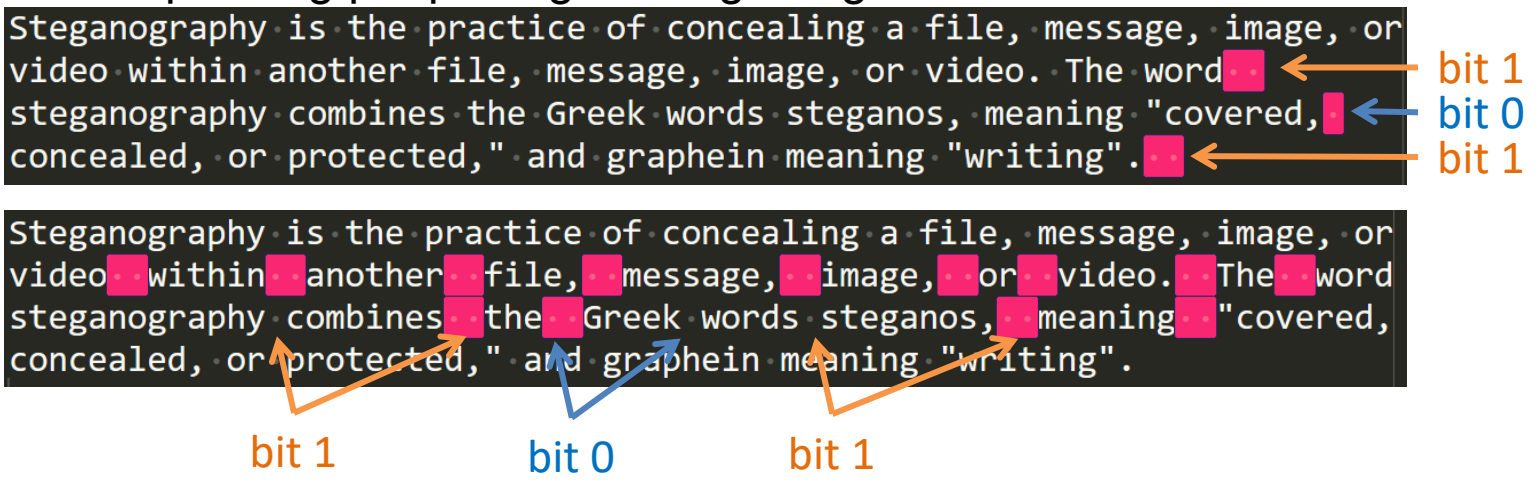
Do sĩ số của lớp đông nên cách đánh giá môn học sẽ được điều chỉnh lại như sau:

- **Các bài tập (~~30%~~ 50%)**
  - Ngôn ngữ lập trình: Python
  - Làm trên IPython Notebook
- ~~**Seminar lý thuyết (20%)**~~
- **Thi lý thuyết cuối kỳ (50%)**
  - Trắc nghiệm + tự luận

Buổi trước



- Nhóm phương pháp dùng khoảng trắng



- Nhóm phương pháp dùng cú pháp
- Nhóm phương pháp dùng ngữ nghĩa

Nói thêm về trường hợp file cover text có định dạng khác .txt

Với file cover text có định dạng khác .txt (vd, file word) thì sẽ có nhiều hay ít chỗ có thể nhúng hơn?

Buổi này: ẩn tin mật trên văn bản theo một hướng tiếp cận khác

- Buổi trước: cho sẵn văn bản che chắn và ẩn tin mật vào trong văn bản này
- Buổi này: từ tin mật, phát sinh ra văn bản che chắn có chứa tin mật trong đó
  - Ví dụ: spammimic.com

spammimic.com

Enter your short secret message:

I love u

Encode

Encoded

Your message **I love u** gets encoded into spam

Dear Friend , Especially for you - this red-hot intelligence . If you no longer wish to receive our publications simply reply with a Subject: of "REMOVE" and you will immediately be removed from our mailing list . This mail is being sent in compliance with Senate bill 1623 , Title 9 ; Section 308 ! This is a ligitimate business proposal ! Why work for somebody else when you can become rich as few as 62 months ! Have you ever notice more people than ever are surfing the web plus nobody is getting any younger . Well, now is your chance to capitalize on this . We will help you process your orders within seconds and turn your business into an E-BUSINESS ! The best thing about our system is that it is absolutely risk free for you ! But don't believe us . Mr Ames of Massachusetts tried us and says "My only problem now is where to park all my cars" ! We are licensed to operate in all states ! We beseech you - act now . Sign up a friend and your friend will be rich too ! Thank-you for your serious consideration of our offer !

Decode | Copy to Clipboard

Your message **227 Nguyen Van Cu** gets encoded into spam as:

Dear Colleague , Especially for you - this red-hot news . If you are not interested in our publications and wish to be removed from our lists, simply do NOT respond and ignore this mail ! This mail is being sent in compliance with Senate bill 1916 ; Title 9 , Section 306 . This is not multi-level marketing . Why work for somebody else when you can become rich in 22 months ! Have you ever noticed the baby boomers are more demanding than their parents & more people than ever are surfing the web ! Well, now is your chance to capitalize on this . WE will help YOU increase customer response by 130% plus decrease perceived waiting time by 160% ! You are guaranteed to succeed because we take all the risk . But don't believe us . Ms Simpson who resides in North Carolina tried us and says "I've been poor and I've been rich - rich is better" . We are a BBB member in good standing ! You will blame yourself forever if you don't order now . Sign up a friend and you'll get a discount of 40% . Thank-you for your serious

Ý tưởng chuyển tin mật thành một văn bản vô hại được đề xuất bởi Peter Wayner (1992), dựa trên **cú pháp phi ngữ cảnh CFG (Context-Free Grammar)**

Nội dung sắp trình bày:

- CFG → văn bản
- Nhúng: tin mật + CFG → văn bản chứa tin mật
- Rút trích: văn bản chứa tin mật + CFG → tin mật

CFG (Context-Free Grammar) là gì?

- Một CFG là một tập các luật dùng để phát sinh ra các chuỗi ký tự
- Tập các chuỗi được phát sinh ra bởi một CFG được gọi là ngôn ngữ của CFG đó
- Ví dụ về một CFG
  - **Start → noun verb**
  - **noun** → Alice | Bob
  - **verb** → is sending | is receiving

- Các từ in đậm (**Start**, **noun**, **verb**) được gọi là các nonterminal symbol
- **Start** là một nonterminal symbol đặc biệt, gọi là start symbol
- Các từ không in đậm (Alice, Bob, is, sending, receiving) được gọi là các terminal symbol
- Một luật của CFG có dạng: L → R với L là một nonterminal symbol, R là chuỗi gồm các nonterminal symbol hoặc các terminal symbol hoặc cả 2; ý của luật L → R là L có thể được mở ra thành R
  - Nếu có L → R1, L → R2, ... thì có thể viết gọn là: L → R1 | R2 | ...

Làm sao để phát sinh ra chuỗi từ một CFG?

- Ban đầu, chuỗi phát sinh gồm một nonterminal symbol là start symbol
- Lặp cho đến khi chuỗi phát sinh gồm toàn terminal symbol:
  - Chọn một nonterminal symbol **ns** trong chuỗi phát sinh hiện tại (vd, chọn cái ở phía trái nhất)
  - Tìm luật có **ns** ở vế trái và thay **ns** trong chuỗi phát sinh bằng vế phải của luật
    - Nếu vế phải có nhiều lựa chọn thì có thể chọn ngẫu nhiên

Phát sinh chuỗi từ CFG – ví dụ 1

- Cho CFG
  - **Start** → **noun verb**
  - **noun** → Alice | Bob
  - **verb** → is sending | is receiving
- Phát sinh chuỗi
  - **Start**
  - **noun verb** (luật: **Start** → **noun verb**)
  - Alice **verb** (luật: **noun** → Alice)
  - Alice is receiving (luật: **verb** → **is receiving**)
- Với cách chọn **noun** và **verb** khác sẽ ra được chuỗi khác

Phát sinh chuỗi từ CFG – ví dụ 2

- Cho CFG
  - **Start → expression**
  - **expression → number | expression + expression | expression - expression**
  - **number → digit | numberdigit**
  - **digit** → 0 | 1 | ... | 9
- Phát sinh chuỗi
  - **Start**
  - **expression** (luật: **Start → expression**)
  - **expression + expression** (luật: **expression → expression + expression**)
  - **expression – expression + expression** (luật: **expression → expression - expression**)
  - **number – expression + expression** (luật: **expression → number**)
  - **numberdigit – expression + expression** (luật: **number → numberdigit**)
  - **digitdigit – expression + expression** (luật: **number → digit**)
  - 27 – **expression + expression** (luật: **digit** → 2, **digit** → 7)
  - ...
  - 27 – 9 + 123
- Có bao nhiêu chuỗi có thể được phát sinh từ CFG này?

Nói thêm: tại sao gọi là Context-Free?

Nitish Chandra, studied at Indian Institute of Technology, Bombay
Answered Jul 26, 2014

Originally Answered: What is the meaning of "Context free" in Context free grammar?

Consider the rule

$$A \rightarrow 0A1$$

What this says is "wherever you find A, you can replace it with $0A1$". Now, consider the rule

$$CAB \rightarrow C0A1B$$

This says "You can replace A with $0A1$ only if it is preceded by C and followed by B" Here, it imposes a condition on when A can be replaced with $0A1$. You can apply this rule only if A appears in this particular *context*. Here, 'context' is used as is generally used in normal English.

In the first case, you didn't need any *context* to apply the rule. You can apply it irrespective of the context in which A appears. So, grammars which contain only rules of first kind are called context-free grammars.

Dùng CFG để tạo ra chuỗi văn bản chứa các bit mật như thế nào?

Ý tưởng

- Trong quá trình phát sinh chuỗi từ CFG, với một nonterminal symbol, có thể có nhiều lựa chọn để thay thế → sử dụng các lựa chọn này để nhúng bit
- Ví dụ với luật L → R1 | R2:
  - Bit 0 = R1
  - Bit 1 = R2
- Ví dụ với luật L → R1 | R2 | R3 | R4:
  - 2 bit 00 = R1
  - 2 bit 01 = R2
  - 2 bit 10 = R3
  - 2 bit 11 = R4

Dùng CFG để tạo ra chuỗi văn bản chứa các bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi bit cần nhúng: 1101

Nhúng

Start	1101
noun verb	1101
Barney **verb**	1101
Barney went fishing **where**	1101
Barney went fishing in Minnesota	1101

Dùng CFG để tạo ra chuỗi văn bản chứa các bit mật như thế nào?

Ví dụ 2

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi bit cần nhúng: 11

Nhúng

Start	11
noun verb	11
Barney **verb**	11

Đã hết bit nhúng nhưng chuỗi phát sinh vẫn chưa hoàn thành
→ phải làm sao?

Dùng CFG để tạo ra chuỗi văn bản chứa các bit mật như thế nào?

Ví dụ 2

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi bit cần nhúng: 11

Nhúng

Start	11
noun verb	11
Barney **verb**	1100...

Một cách là tiếp tục nhúng một bit 1 và các bit 0 cho đến khi chuỗi phát sinh được hoàn thành
Khi rút trích, ta sẽ ra được chuỗi bit có đuôi 100...
và có thể dễ dàng cắt đuôi này đi

Dùng CFG để tạo ra chuỗi văn bản chứa các bit mật như thế nào?

Ví dụ 2

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi bit cần nhúng: 11

Nhúng

Start	11
noun verb	11
Barney **verb**	1100...
Barney went bowling **where**	1100...
Barney went bowling in Iowa	1100...

Dùng CFG để tạo ra chuỗi văn bản chứa các bit mật như thế nào?

Ví dụ 3

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney | Mary
- **verb** → … | went bowling **where**

noun có 5 lựa chọn
→ nhúng được mấy bit?

Dùng CFG để tạo ra chuỗi văn bản chứa các bit mật như thế nào?

Ví dụ 3

Cho CFG

- **Start → noun verb**
- **noun** → Alice | Bob | Fred | Barney | Mary
- **verb** → [illegible]vent bowling **where**

Một cách là chỉ dùng 4 lựa chọn đầu và nhúng 2 bit

Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ý tưởng

- Cần có CFG
- Bài toán cần giải là tìm đường đi từ start symbol đến chuỗi văn bản chứa bit mật; một khi đã tìm ra được đường đi này, ta có thể dễ dàng biết được các bit mật đã được nhúng
- Tìm đường đi như thế nào?
  - Một cách là dùng DFS (Depth First Search)

Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”

Start

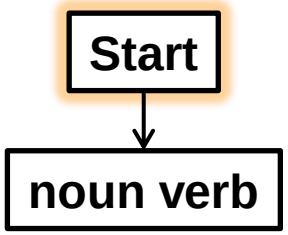
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”



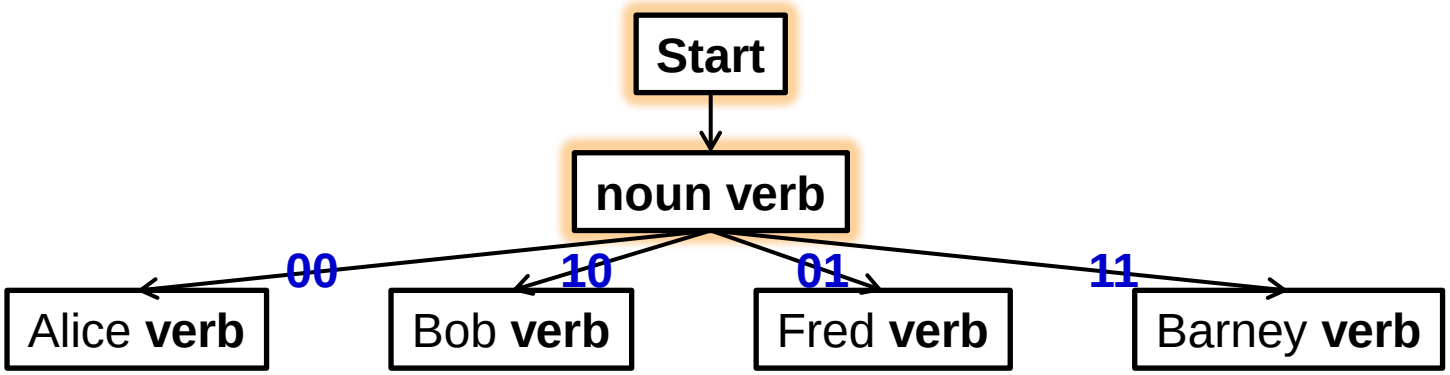
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”



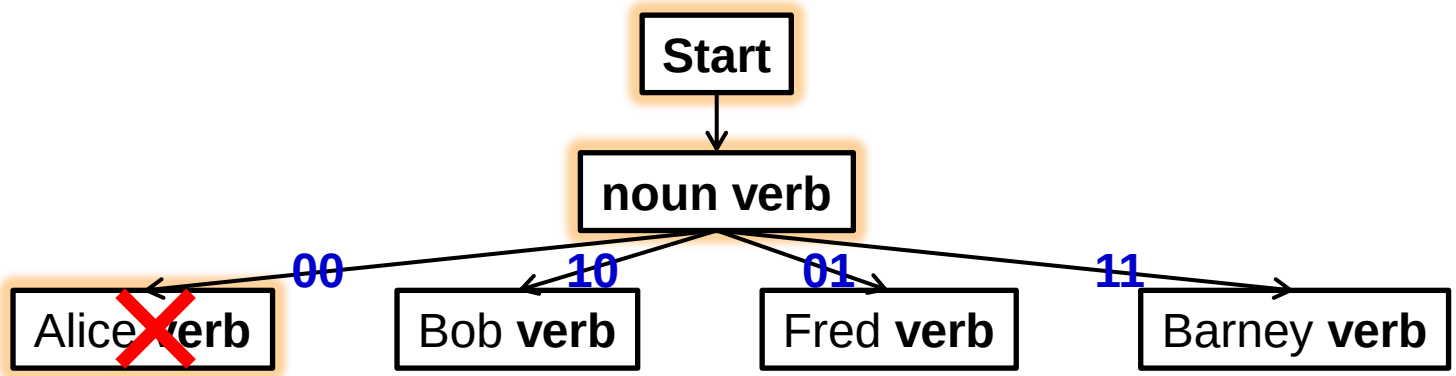
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”



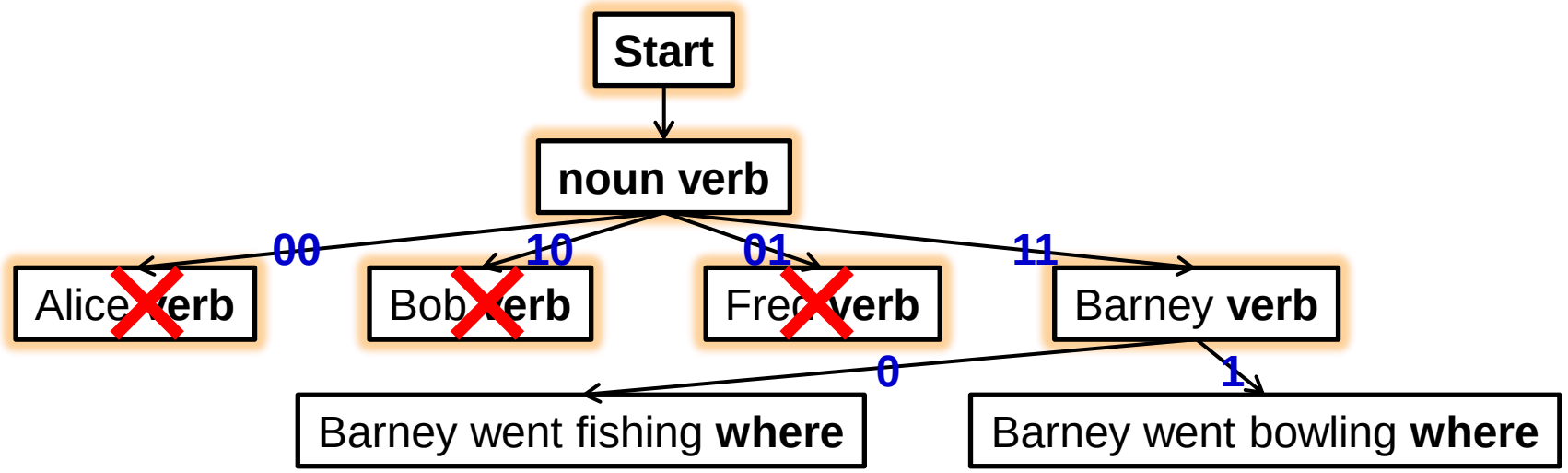
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”



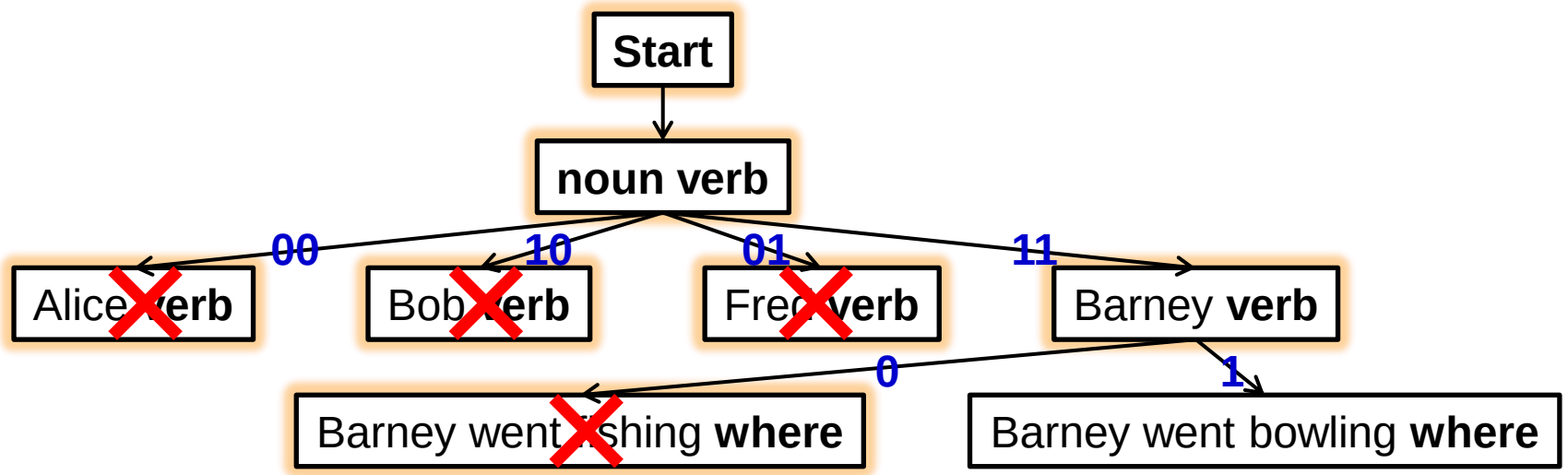
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”



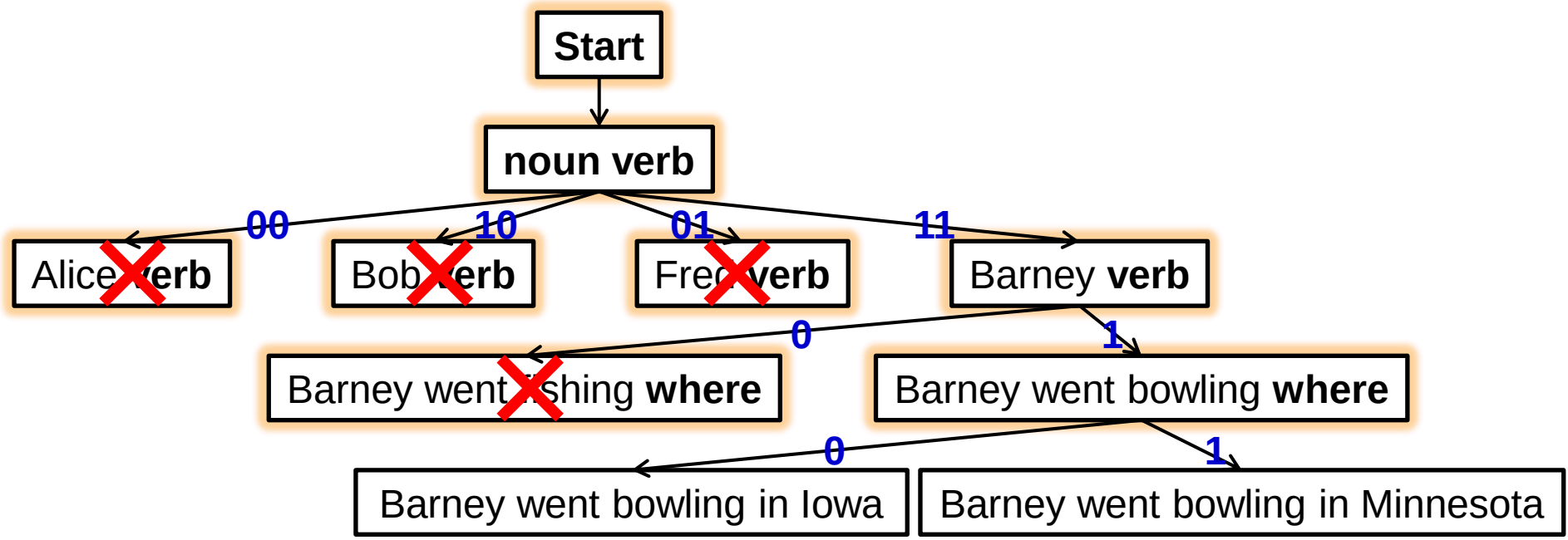
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”



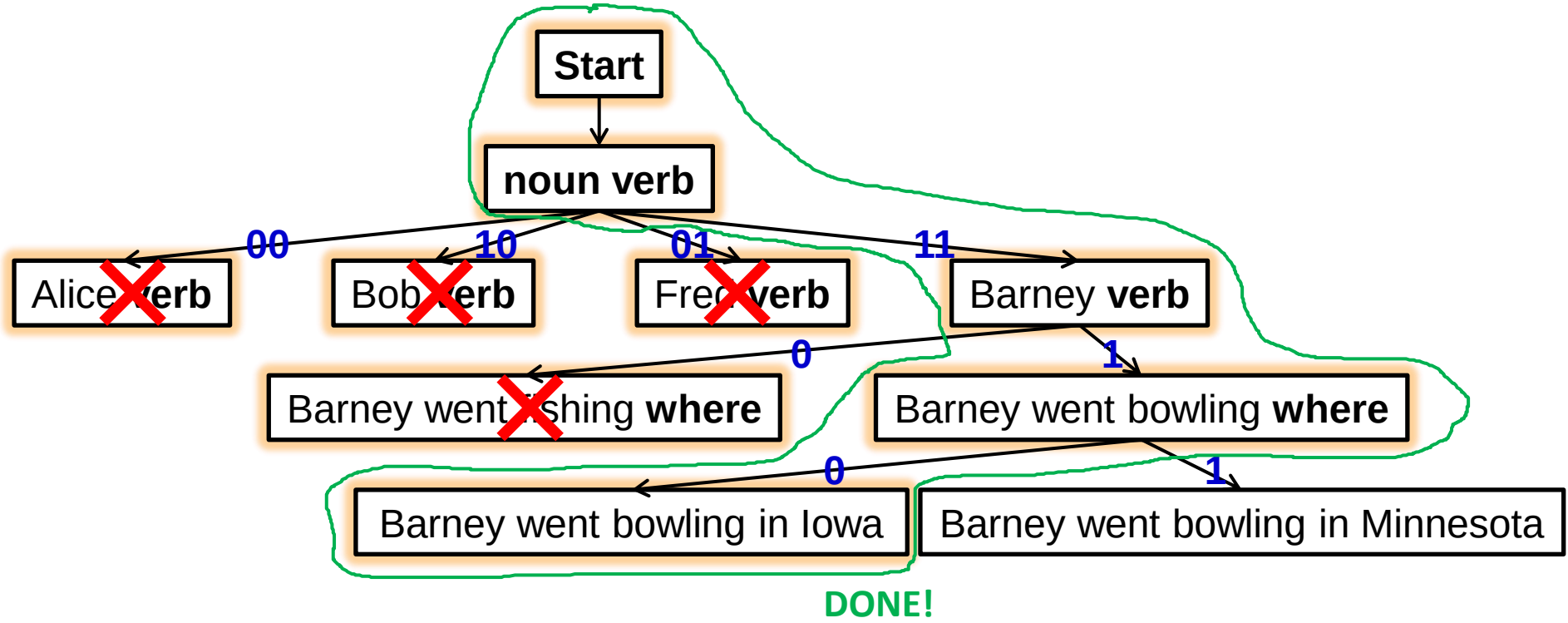
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 1

Cho CFG

- **Start** → **noun verb**
- **noun** → Alice | Bob | Fred | Barney
- **verb** → went fishing **where** | went bowling **where**
- **where** → in Iowa | in Minnesota

Cho chuỗi văn bản chứa bit mật: “Barney went bowling in Iowa”



Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 2

Cho CFG

- **Start** → **name action** | **whobe where**
- **name** → Alice | Bob
- **action** → is here | is there
- **whobe** → Alice is | Bob was
- **where** → here | there

Nhúng chuỗi bit 101:

- **Start** 101
- **whobe where** 101
- Alice is **where** 101
- Alice is there 101

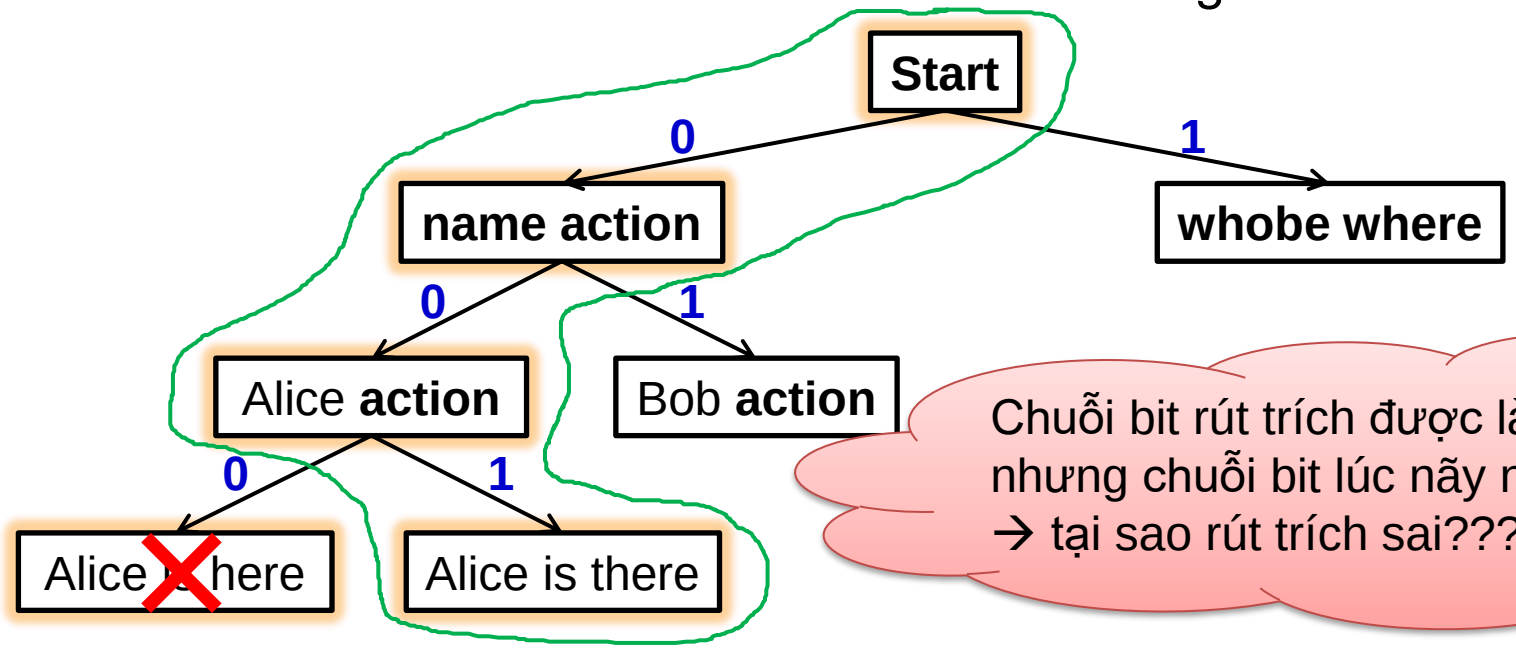
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 2

Cho CFG

- **Start** → **name action** | **whobe where**
- **name** → Alice | Bob
- **action** → is here | is there
- **whobe** → Alice is | Bob was
- **where** → here | there

Rút trích từ chuỗi “Alice is there”: xem trên bảng ...



Chuỗi bit rút trích được là **001**,
nhưng chuỗi bit lúc nãy nhúng là **101**
→ tại sao rút trích sai???

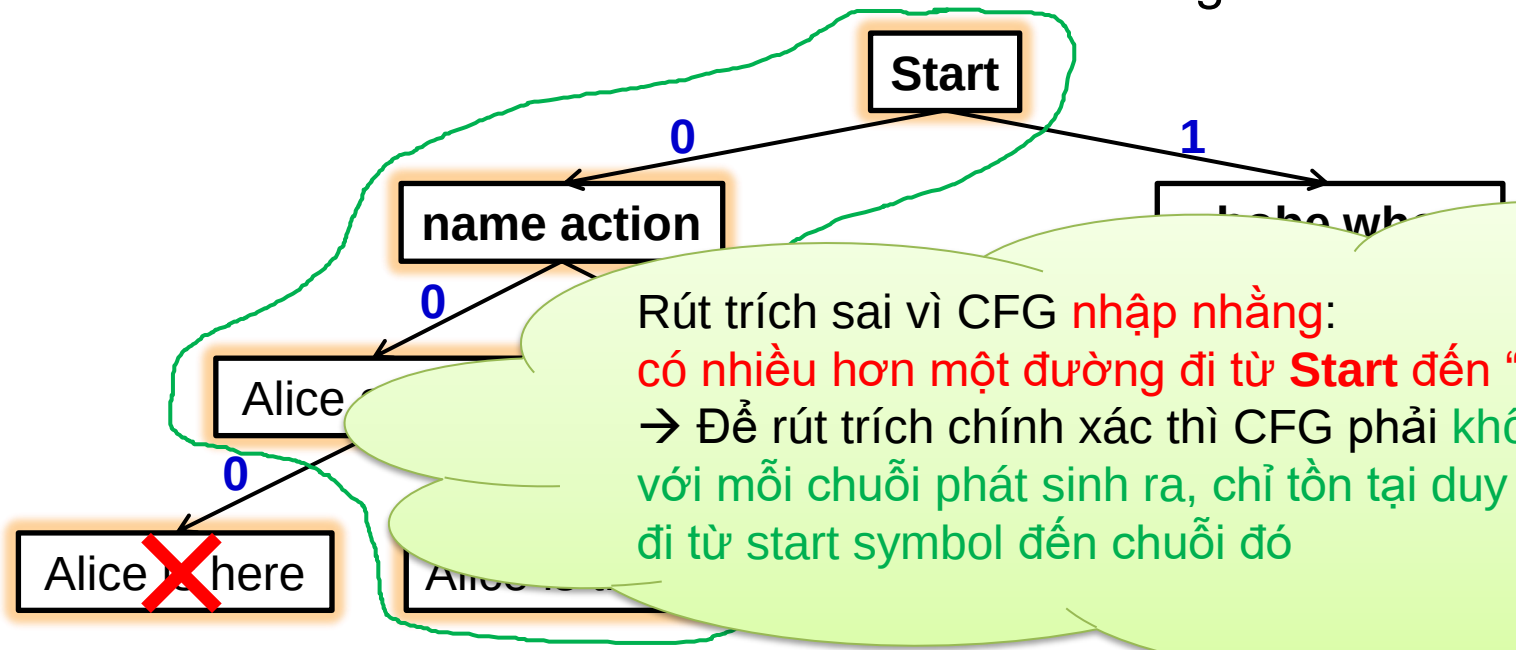
Rút trích bit mật từ chuỗi văn bản chứa bit mật như thế nào?

Ví dụ 2

Cho CFG

- **Start** → **name action** | **whobe where**
- **name** → Alice | Bob
- **action** → is here | is there
- **whobe** → Alice is | Bob was
- **where** → here | there

Rút trích từ chuỗi “Alice is there”: xem trên bảng ...



Rút trích sai vì CFG nhập nhằng:
có nhiều hơn một đường đi từ **Start** đến “Alice is there”
→ Để rút trích chính xác thì CFG phải không nhập nhằng:
với mỗi chuỗi phát sinh ra, chỉ tồn tại duy nhất một đường đi từ start symbol đến chuỗi đó

Quiz 1

- [ ] 0100110

Start → **adjective noun tense verb**
adjective → the **size** | a **size**
size → tiny | small | large | big
noun → saw | ladder | truth | boy
tense → is | was
verb → waiting | standing

Quiz 2

- “Alice sent email to all relatives.”

Start → **noun verb**
noun → Alice | Bob
verb → sent mail **to** | sent email **to**
to → to **rel recipient**
rel → all | some
recipient → friends | relatives

Nhận xét về phương pháp ẩn tin mật trên văn bản bằng CFG

Để đạt tính vô hình cao và sức chứa lớn thì phải tốn nhiều thời gian ngồi thiết kế CFG ☹

Tài liệu tham khảo

Salomon, Data privacy and security: encryption and information hiding – chapter 10, Springer Science & Business Media (2003)