Compilers Techniques
Lecture 6
المحاضرة السادسة

Syntax Analyser

المحلل القواعدي

Reference: Aho2 Sections 3.6-3.7; Aho1 pp. 113-125; Grune 2.1.6.1-2.1.6.6 (different style); Hunter 3.3 (very condensed); Cooper1 2.4-2.4.3

السنة الرابعة – المستوى السابع - الهندسة المعلوماتية

Lectures Topics

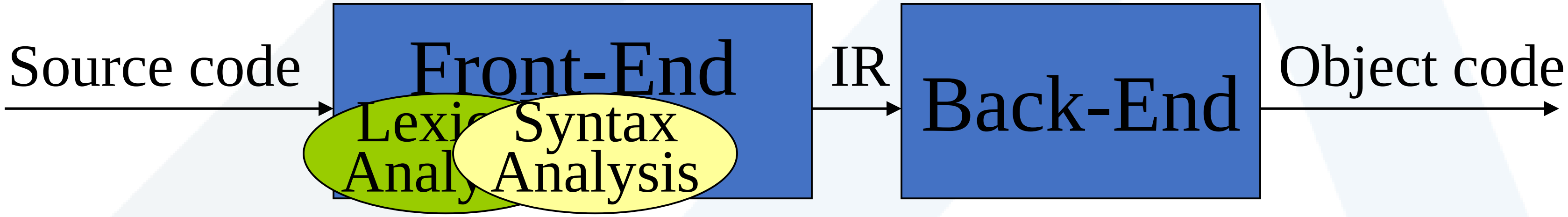
محاضرة اليوم التحليل القواعدي Syntax Analysis

قواعد حرّة السياقcontext-free grammars

الاشتقاق derivations

شجرة الإعراب parse trees

الغموض ambiguity.



لا يمكننا وصف جميع اللغات بالتعابير النظامية

التحليل القواعدي Syntax Analysis

القوة الوصفية للتعابير النظامية لها حدود:

- لا تستطيع REs وصف البنى المضمنة أو المتوازنة مثل: جميع سلاسل الأقواس الموازنة {(), (()), ((())), ...} ، أو مجموعة جميع الأصفار التي يليها عدد مساوي من الواحدات {01, 0011, 000111, ...} .
- لا يمكننا في التعابير النظامية استخدام رمز غير تحديدي Non Terminal قبل أن يعرّف بشكل نهائي وتام.

هرمية شومسكي في الإعراب Chomsky's hierarchy of Grammars:

- 1-ضبط بنى الصيغ النحوية Phrase structured.
- 2-الحساسية للسياق Context Sensitive.

  عدد رموز الطرف الأيسر أقل أو يساوي عدد رموز الطرف الأيمن.
- 3-النحو حر السياق Context-Free

  الطرف الأيسر هو دوماً رمز غير تحديدي non-terminal.
- 4-اللغات القياسية Regular Language.

  القواعد المسموحة هي فقط القواعد التي من الشكل $A\rightarrow\varepsilon$, $A\rightarrow a$, $A\rightarrow pB$.

Regular Languages $\subset$ Context-Free Languages $\subset$ Cont.Sens.Ls $\subset$ Phr.Str.Ls

- **<u>الإعراب حر السياق معرف فقط بالقواعد حرة السياق.</u>**

يعبر عن القواعد G، برباعية 4-tuple G={S,N,T,P}، حيث:

S هو رمز البداية
N هي مجموعة من الرموز غير التحديدية non-terminal symbols
T مجموعة من الرموز التحديدية terminal symbols
P مجموعة من قواعد الإنتاج production rules.

- مثال:

CatNoise→CatNoise miau	**rule 1**
| miau	**rule 2**

- يمكننا استخدام قواعد CatNoise لتوليد جمل مثل:

Rule	**Sentential Form**
-	*CatNoise*
1	*CatNoise miau*
2	*miau miau*

- يطلق على إعادة الكتابة السابقة تسمية الاشتقاق derivation.

تسمى عملية اكتشاف الاشتقاق لبعض الجمل الإعراب parsing!

الاشتقاق Derivation: هو سلسلة من مراحل الاشتقاق:

- في كل خطوة، نختار رمز غير تحديدي Non-Terminal للاستبدال.
- يمكن أن تقود الاختيارات المختلفة لاشتقاقات مختلفة.

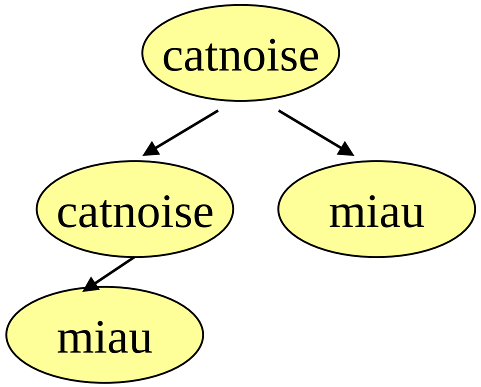
هنالك طريقتان للاشتقاق:

- من اليسار :Leftmost derivation
- نقوم في كل مرحلة باستبدال الرمز غير التحديدي الأيسر.

- من اليمين :Rightmost derivation
- نقوم في كل مرحلة باستبدال الرمز غير التحديدي الأيمن.

(لن نهتم بالاشتقاقات العشوائية)

شجرة الإعراب A parse tree: هي تمثيل بياني لاشتقاق يرشح الاختيار وفقاً لتتالي الاستبدال.



البناء Construction:

نبدأ برمز البداية (جذر الشجرة)،

من أجل كل شكل للجملة:

نضيف العقد الأبناء لكل صيغة جملة sentential form:

- *نضيف العقد الأبناء (لكل رمز على الجهة اليمنى من قواعد الإنتاج التي تم تطبيقها) إلى العقدة المتوافقة مع الرمز اليساري)*
- *فتتشكل الجملة مقروءةً من اليسار إلى اليمين كأوراق للشجرة.*

أوجد الاشتقاق من اليمين ومن اليسار وشجرة الإعراب للتعبير x-2*y

1. Goal → Expr
2. Expr → Expr op Expr
3. | number
4. | id
5. Op → +
6. | -
7. | *
8. | /

- الاشتقاق من اليسار ومن اليمين قد يؤدي إلى أشجار إعراب مختلفة.
- مثلاً في المثال السابق يمكن تنفيذ اشتقاقين:
  - الأول سيقيم كـ$x - (2*y)$
  - الثاني كـ$(x - 2)*y$
- ليس هنالك أية ملاحظة حول الأسبقيات (أو أمر مضمن حول التقويم)
- **<u>ولإضافة الأسبقيات:</u>** نجبر المعرب على تمييز التعابير الفرعية ذات الأسبقية الأعلى أولاً.

القواعد التى تنتج أكثر من شجرة إعراب لجملة ما تعتبر غامضة، أو:

- إذا كانت للقواعد أكثر من اشتقاق من اليسار إلى اليمين من أجل صيغة جملة واحدة، فإننا نقول عن القواعد غامضة.
- إذا كان للقواعد أكثر من اشتقاق من اليمين إلى اليسار من أجل صيغة جملة واحدة فإننا نقول عن القواعد أنها غامضة.

مثال:

- Stmt → if Expr then Stmt |
  if Expr then Stmt else Stmt | …other…

- ماهو اشتقاق:

  - if E1 then if E2 then S1 else S2
  - Stmt → if Expr then **Stmt** → if Expr then if Expr Stmt else Stmt

القواعد التى تنتج أكثر من شجرة إعراب لجملة ما تعتبر غامضة، أو:

مثال:

- Stmt → if Expr then Stmt |

  if Expr then Stmt else Stmt | …other…

- ماهو اشتقاق:

  - if E1 then if E2 then S1 else S2

Method1:

Stmt → if Expr then **Stmt**

→ if Expr then if Expr then Stmt else Stmt

Method2:

Stmt → if Expr then **Stmt** else Stmt

→ if Expr then if Expr then Stmt else Stmt

- إعادة كتابة القواعد لتجنب الغموض.
- مطابقة كل حالة else لـ if الداخلية التي لم تطابق.

1. Stmt → IfwithElse
2. | IfnoElse
3. IfwithElse → if Expr then IfwithElse else IfwithElse
4. | … other stmts…
5. IfnoElse → if Expr then Stmt
6. | if Expr then IfwithElse else IfnoElse

Stmt → if Expr then Stmt |
if Expr then Stmt else Stmt | …other…

(1)	Stmt
(2)	IfnoElse
(5)	if Expr then Stmt
(?)	if E1 then Stmt
(1)	if E1 then IfwithElse
(3)	if E1 then if Expr then IfwithElse else IfwithElse
(?)	if E1 then if E2 then IfwithElse else IfwithElse
(4)	if E1 then if E2 then S1 else IfwithElse
(4)	if E1 then if E2 then S1 else S2

الغموض العميق Deeper Ambiguity

- يشير الغموض إلى تشوش في CFG.
- يمكن أن يسبب التحميل الزائد غموضاً أعمق.
  - مثلاً: في a=b(3) قد تكون b إما تابع أم متغير.
- يتطلب التخلص من الغموض أعلاه السياق التالي:
  - تحديد نمط لا يخضع لقواعد حرة السياق
  - تحتاج لقيم التصريحات.
  - تحتاج حل قواعدي إضافي.
- حل مسألة الغموض Resolving ambiguity:
  - في حال أننا نعمل وفق القواعد حرة السياق: نعيد كتابة القواعد.
  - الغموض في القواعد الحساسة للسياق context-sensitive ambiguity: تختبر بوسائل أخرى، وتحتاج إلى معارف حول الأنماط وتصاريح وهي مسألة تصميم اللغة.
- يقبل أحياناً كاتب المترجم قواعد غامضة: حيث تقوم وسائل الإعراب بعملية التصحيح

- المعربات من الأعلى إلى الأسفل Top-down parsers:
  - تشكيل العقدة الأعلى في الشجرة وبعدها تشكيل العقد الباقية بهيكلية معرفة مسبقاً (مثل العمق أولاً (depth-first
  - نختار قاعدة إنتاج ونحاول مطابقة الدخل معها؛ فإذا حصل خطأ نتراجع .backtrack
  - الاشتقاق عادةً من اليسار إلى اليمين لسلسلة الدخل (نمسحها من اليسار إلى اليمين).
  - بعض القواعد لا نحتاج التراجع فيها backtrack-free **(الإعراب التنبؤي predictive parsing).**
- المعربات من الأسفل إلى الأعلى Bottom-up parsers:
  - نبني الشجرة لسلسلة دخل ابتداءً من الأوراق ونسعى للوصول إلى السطح (الجذر).
  - يستخدم الإعراب من العمق إلى السطح المسح من اليسار إلى اليمين ويحاول بناء اشتقاق من اليمين إلى اليسار تراجعياً.
  - يعالج صفاً كبيراً من القواعد.

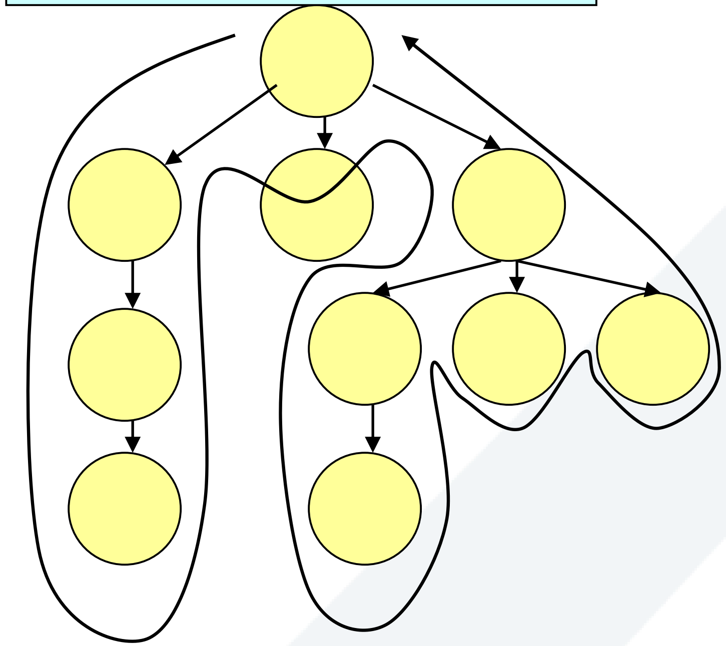
Parsing تقنيات الإعراب techniques

المحلل القواعدي Syntax Analysis

يشبه الإعراب Top-Down حالتين خاصتين من التجوال من خلال البحث في العمق أولاً:

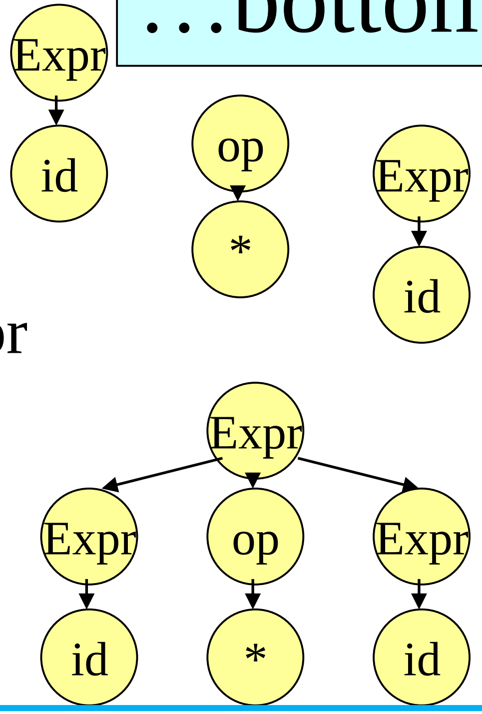
- الترتيب المسبق Pre-order: يجول على العقدة node x بعدها يجول على أشجار x’ الفرعية من اليسار إلى اليمين (وتطبق الفعالية عندما نزور العقدة)
- الترتيب اللاحقPost-order: يزور أولاً أشجار x’ الفرعية من اليسار إلى اليمين وبعدها العقدة x. (تطبق العملية قبل تركنا العقدة للمرة الأخيرة)

Top-Down



...bottom-up!

id * id
Expr * id
Expr op id
Expr op Expr
Expr



Syntax Analysis المحلل القواعدي

1-بناء الجذر برمز **البداية** المحدد بالقواعد.

2-تكرار الخطوات الآتية حتى ينطبق تتالي شجرة الإعراب مع سلسلة الدخل.

- بفرض أن العقدة المسماة A، تقع على الجانب الأيسر، نختار قاعدة إنتاج ترتبط بـA على الطرف الأيسر ونبني من أجل كل رمز من الرموز الموجودة على الجانب الأيمن أبناء العقدة الموافقين.
- عند إضافة رمز تحديدي للتتالي وفي حال عدم مطابقة هذا الرمز مع التتالي المرغوب نتراجع .backtrack
- نبحث في العقدة التالية التي يتوجب توسيعها.

المفتاح هو انتقاء الجزء اليميني من قاعدة الإنتاج في الخطوة الأولى، بحيث يقاد الاختيار من خلال سلسلة الدخل.

مثال:

1. *Goal → Expr*
2. *Expr → Expr + Term*
3. *| Expr – Term*
4. *| Term*
5. *Term → Term * Factor*
6. *| Term / Factor*
7. *| Factor*
8. *Factor → number*
9. *| id*

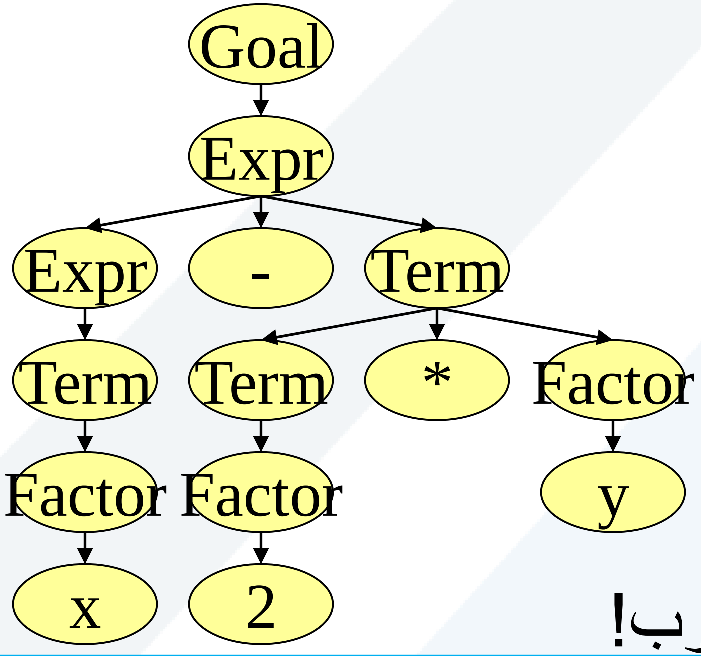
الإعراب من الأعلى إلى الأسفل بالتراجع التدريجي
Top-Down Recursive-Descent Parsing

Syntax Analysis المحلل القواعدي

Rule	Sentential Form	Input
-	*Goal*	| x – 2*y
1	*Expr*	| x – 2*y
2	*Expr + Term*	| x – 2*y
4	*Term + Term*	| x – 2*y
7	*Factor + Term*	| x – 2*y
9	*id + Term*	| x – 2*y
Fail	*id + Term*	x | – 2*y
Back	*Expr*	| x – 2*y
3	*Expr – Term*	| x – 2*y
4	*Term – Term*	| x – 2*y
7	*Factor – Term*	| x – 2*y
9	*id – Term*	| x – 2*y
Match	*id – Term*	x – | 2*y
7	*id – Factor*	x – | 2*y
9	*id – num*	x – | 2*y
Fail	*id – num*	x – 2 | *y
Back	*id – Term*	x – | 2*y
5	*id – Term * Factor*	x – | 2*y
7	*id – Factor * Factor*	x – | 2*y
8	*id – num * Factor*	x – | 2*y
match	*id – num * Factor*	x – 2* | y
9	*id – num * id*	x – 2* | y
match	*id – num * id*	x – 2*y |

مثال: إعراب *x-2*y*

1. *Goal → Expr*
2. *Expr → Expr + Term*
3. *| Expr – Term*
4. *| Term*
5. *Term → Term * Factor*
6. *| Term / Factor*
7. *| Factor*
8. *Factor → number*
9. *| id*



Rule	Sentential Form	Input
-	*Goal*	| x – 2*y
1	*Expr*	| x – 2*y
2	*Expr + Term*	| x – 2*y
2	*Expr + Term + Term*	| x – 2*y
2	*Expr + Term + Term + Term*	| x – 2*y
2	*Expr + Term + Term + … + Term*	| x – 2*y

يمكن اختيار احتمالات أخرى للتوسيع

- الخيار الخاطئ يقود إلى عدم التحديد!
- وهي إحدى الخصائص غير الجيدة لهذا المعرب!
- يجب أن يأخذ المعرب الخيار الصحيح!