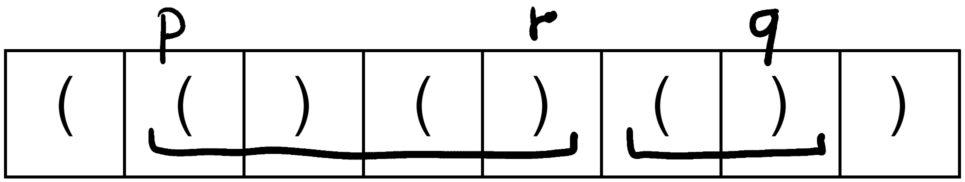
Earley's Algorithm

J. Earley, "An efficient context-free parsing algorithm", *Communications of the Association for Computing Machinery*, **13**:2:94-102, 1970.

CYK vs Earley’s Parser Comparison

Z ::= X Y Z parses w_{pq}

- CYK: if d_{pr} parses X and $d_{(r+1)q}$ parses Y, then in d_{pq} stores symbol Z
- Earley’s parser:
  in set S_q stores *item* (Z ::= XY. , p)
- Move forward, similar to top-down parsers
- Use dotted rules to avoid binary rules



Example: expressions

D ::= e EOF

e ::= ID | e – e | e == e

Rules with a dot inside

D ::= . e EOF | e . EOF | e EOF .

e ::= . ID | ID .
| . e – e | e . – e | e – . e | e – e .
| . e == e | e . == e | e == . e | e == e .

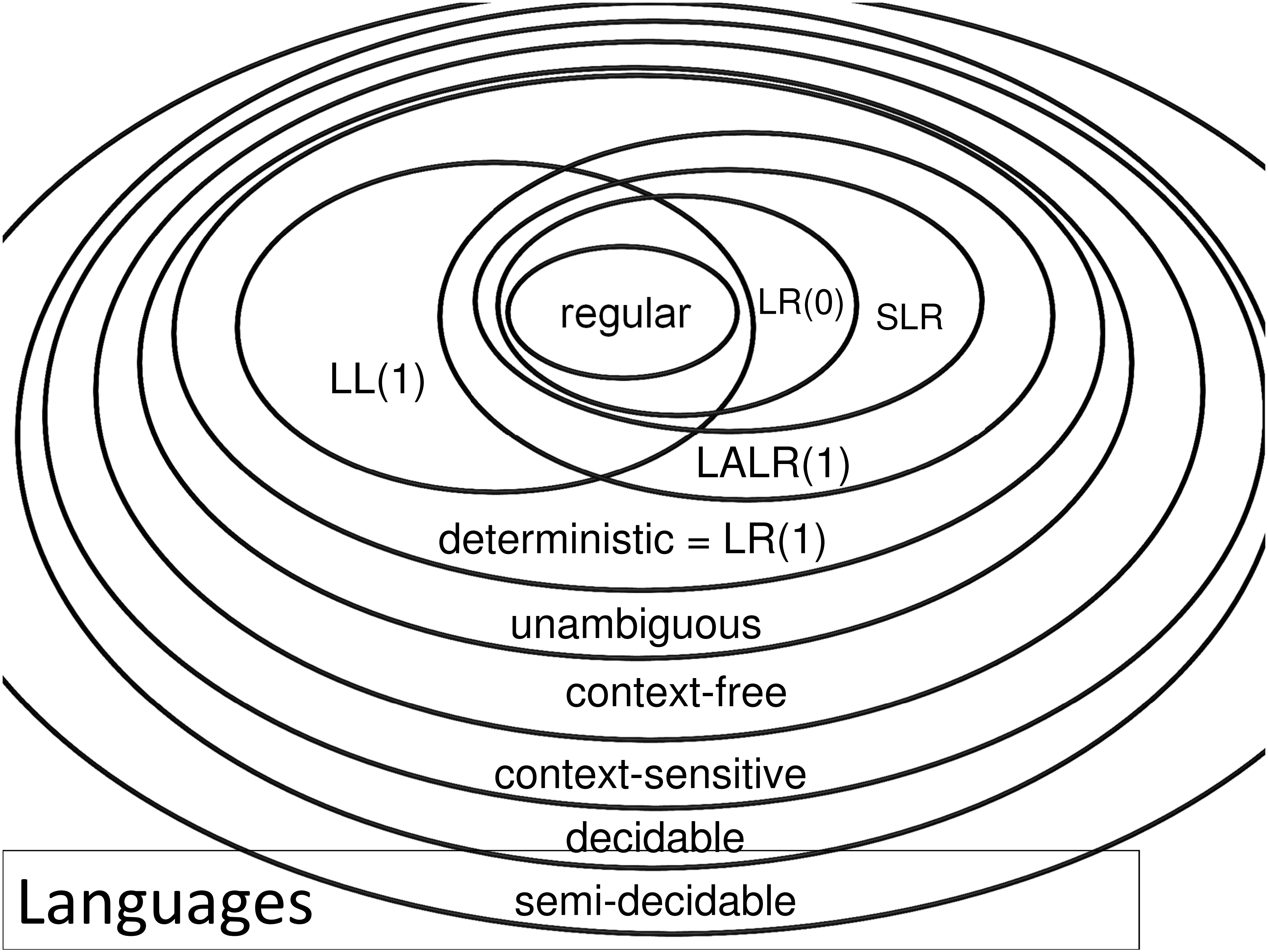
		ID s1	- s2	ID s3	==	ID	EOF
	ε .e EOF .ID .e-e .e=e	ID ID. e.EOF e.-e e.=e	ID- e-.e	ID-ID e-e. e.EOF e.-e e.=e	ID-ID== e=.e	ID-ID==ID e=e. e-e.	e.EOF
ID		ε	-	-ID	-ID==	-ID==ID	
-			ε .ID .e-e .e=e	ID ID. e.-e e.=e	ID== e=.e	ID==ID e=e.	
ID				ε	==	==ID	
==					ε .ID .e-e .e=e	ID ID. e.-e e.=e	
ID						ε	
EOF							ε

e ::= . ID | ID .
| . e – e | e . – e | e – . e | e – e .
| . e == e | e . == e | e == . e | e == e .

Remark: Grammars and Languages

- Language S is a set of words
- For each language S, there can be multiple possible grammars G such that S=L(G)
- Language S is
  - Non-ambiguous if there exists a non-ambiguous grammar for it
  - LL(1) if there is an LL(1) grammar for it
- Even if a language has ambiguous grammar, it can still be non-ambiguous if it also has a non-ambiguous grammar

Languages


regular
LR(0)
SLR
LL(1)
LALR(1)
deterministic = LR(1)
unambiguous
context-free
context-sensitive
decidable
semi-decidable