LR 構文解析

アウトライン

- LR構文解析
- LR(0)
  - LR(0)オートマトンによるshift/reduceの判断法
  - LR(0)アルゴリズムの実際
  - 形式言語理論の立場からみたLR(0)の仕組み
- LR(0)の改良版
  - SLR
  - LR(k)
  - LALR

LR 構文解析

- ボトムアップ解析の一種
  - L (Left-to-right): 入力を左から右に走査
    （cf. CKY　パージング）
  - R(Rightmost derivation): 語の最右導出に対応
    e.g. 文法 G = {E → T | E + T,　　T → x | T*x}
    E → E+T → E+T*x → E+x*x → T+x*x → x+x*x
    逆から読むとLRの解析過程に相当

- LR(k), SLR, LALR(1)などのバリエーションあり
- LR(k)の方がLL(k)より広い文法を扱える（e.g. 左再帰）

LR解析の流れ

- 各時点で以下の情報を保持
  - スタックΓ : すでに認識したシンボル列を保持
  - 残りの入力w
- 初期状態(ε, w)から以下の操作を繰り返す
  - shift: 入力を読んでスタックに移動

    $(\Gamma, aw) \to (\Gamma a, w)$
  - reduce: 規則に従ってスタック上のシンボルを還元

    $(\Gamma X_1 \ldots X_k, w) \to (\Gamma A, w) \quad \text{if } A \to X_1 \ldots X_k$

例： $S' \to S\$$　$S \to (L)$　$S \to x$　$L \to S$　$L \to L,S$

Shift: 入力を読んでスタックに移動

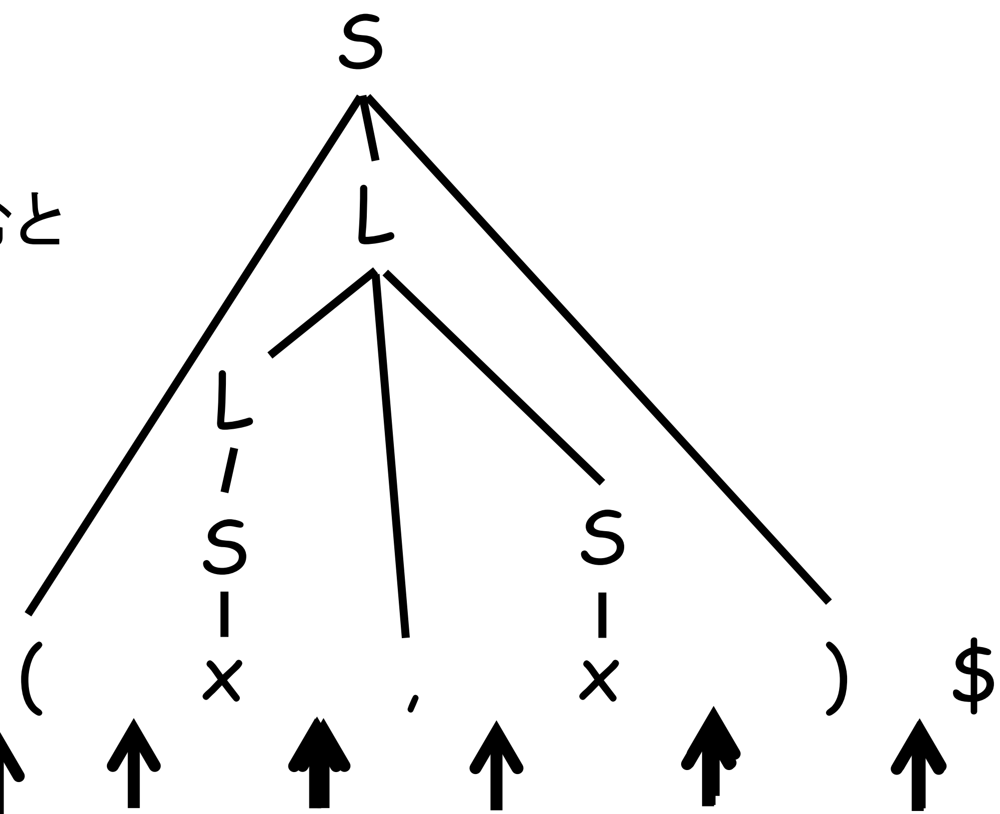
$(\Gamma, aw) \to (\Gamma a, w)$

Reduce: 規則に従ってスタック上の シンボルを還元

$(\Gamma X_1 \ldots X_k, w) \to (\Gamma A, w)$　if $A \to X_1 \ldots X_k$

スタック	入力
	(x,x)$
(	x,x)$
(x	,x)$
(S	,x)$
(L	,x)$
(L,	x)$
(L,x	)$
(L,S	)$
(L	)$
(L)	$
S	$

逆から読むと
最右導出



問題点： shift/reduce の選択は？

S'→S$ S→(L) S→x L→S L→L,S

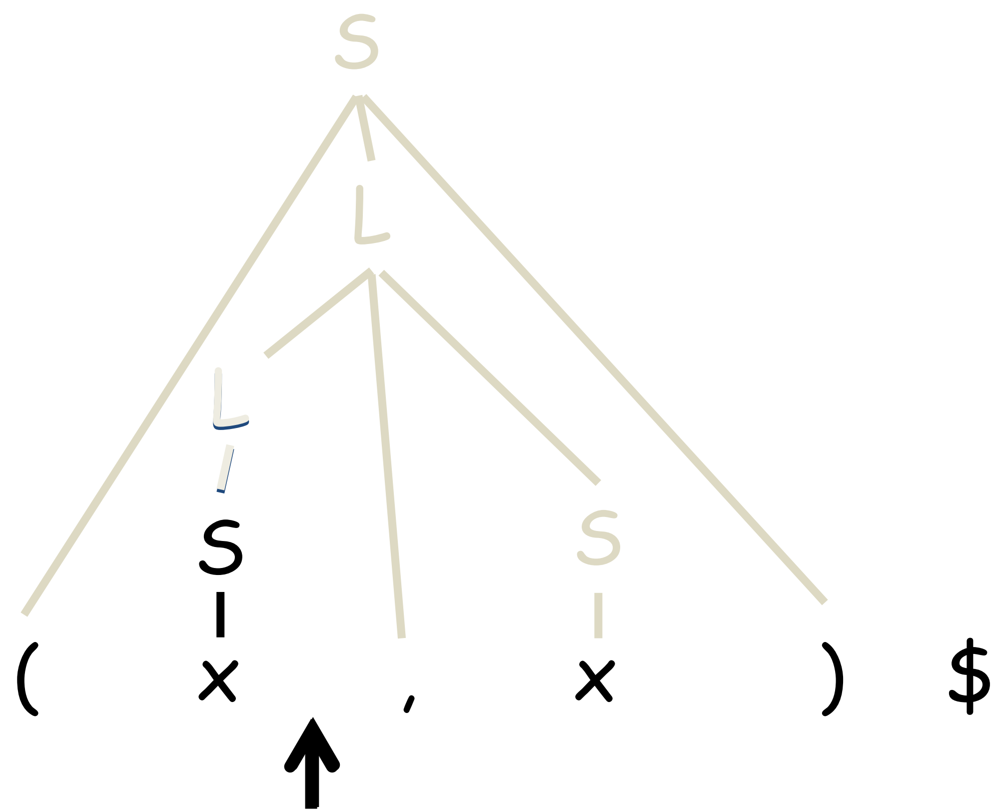
Shift: 入力を読んでスタックに移動

$(\Gamma, aw) \to (\Gamma a, w)$

Reduce: 規則に従ってスタック上の シンボルを還元

$(\Gamma X_1 \ldots X_k, w) \to (\Gamma A, w)$ if $A \to X_1 \ldots X_k$

スタック	入力
	(x,x)$
(	x,x)$
(x	,x)$
(S	,x)$
(L	x)$



問題点： shift/reduce の選択は？

$S' \to S\$ \quad S \to (L) \quad S \to x \quad L \to S \quad L \to L,S$

スタック	入力
	(x,x)$
(	x,x)$
(x	,x)$
(S	,x)$
(S,	x)$
(S, x	)$
(S, S	)$
(S, S)	$

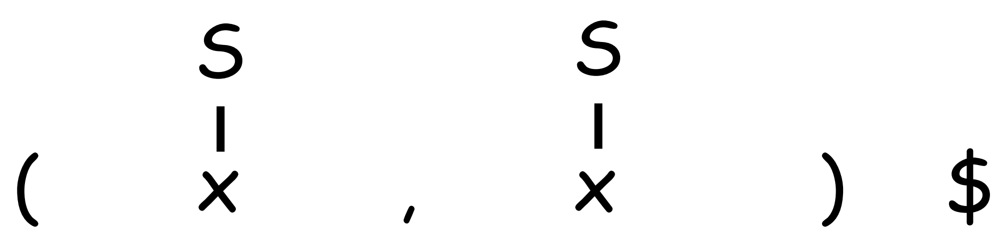
Shift: 入力を読んでスタックに移動

$(\Gamma, aw) \to (\Gamma a, w)$

Reduce: 規則に従ってスタック上の シンボルを還元

$(\Gamma X_1 \ldots X_k, w) \to (\Gamma A, w)$ if $A \to X_1 \ldots X_k$

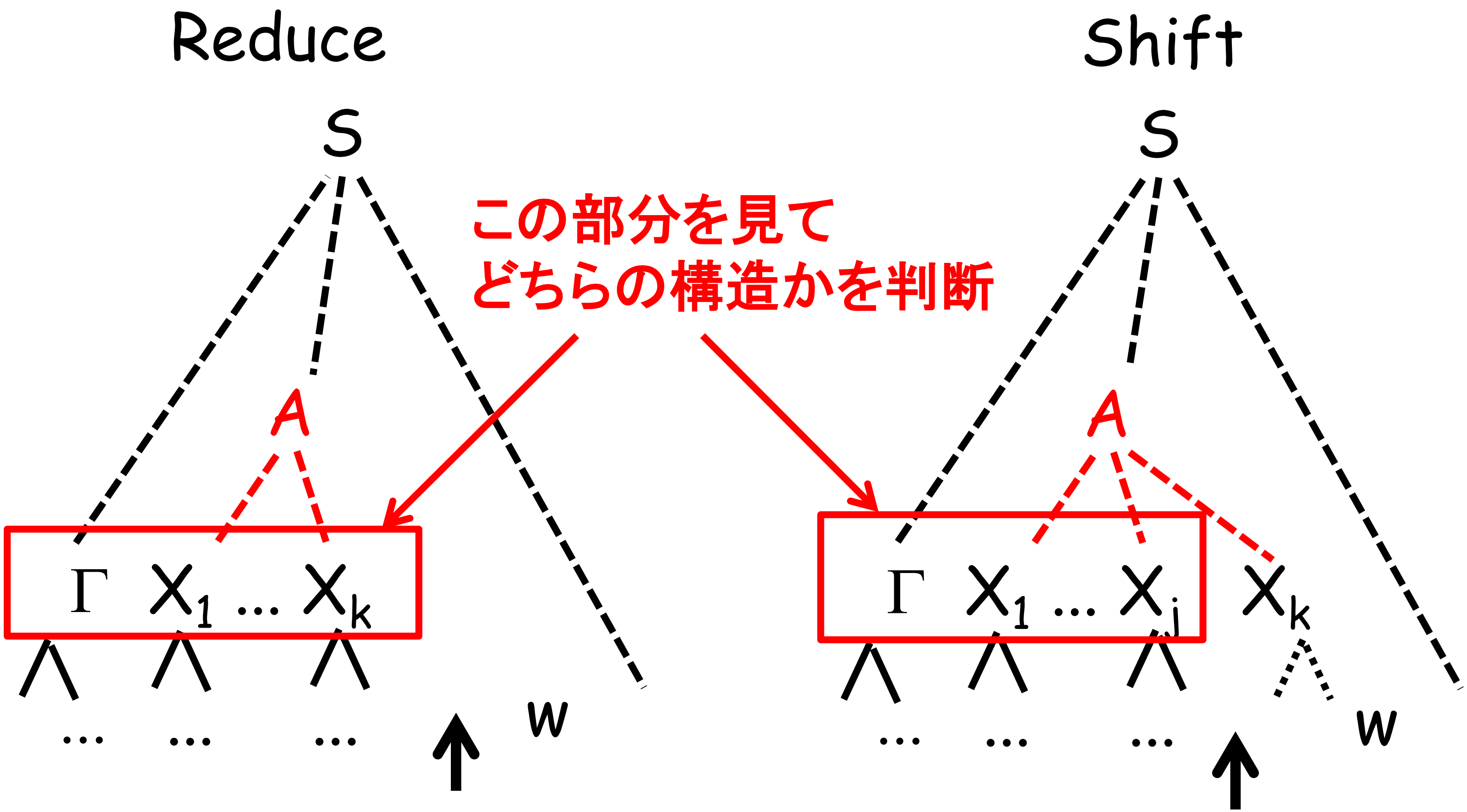
shift/reduce の選択方法によって
LR(0), SLR, LR(k)
などのバリエーション



アウトライン

- LR構文解析
- LR(0)
  - shift/reduceの判断法
  - LR(0)オートマトン
  - LR(0)アルゴリズムの実際
  - 形式言語理論の立場からみたLR(0)の仕組み
- LR(0)の改良版
  - SLR
  - LR(k)
  - LALR

Shift vs Reduce



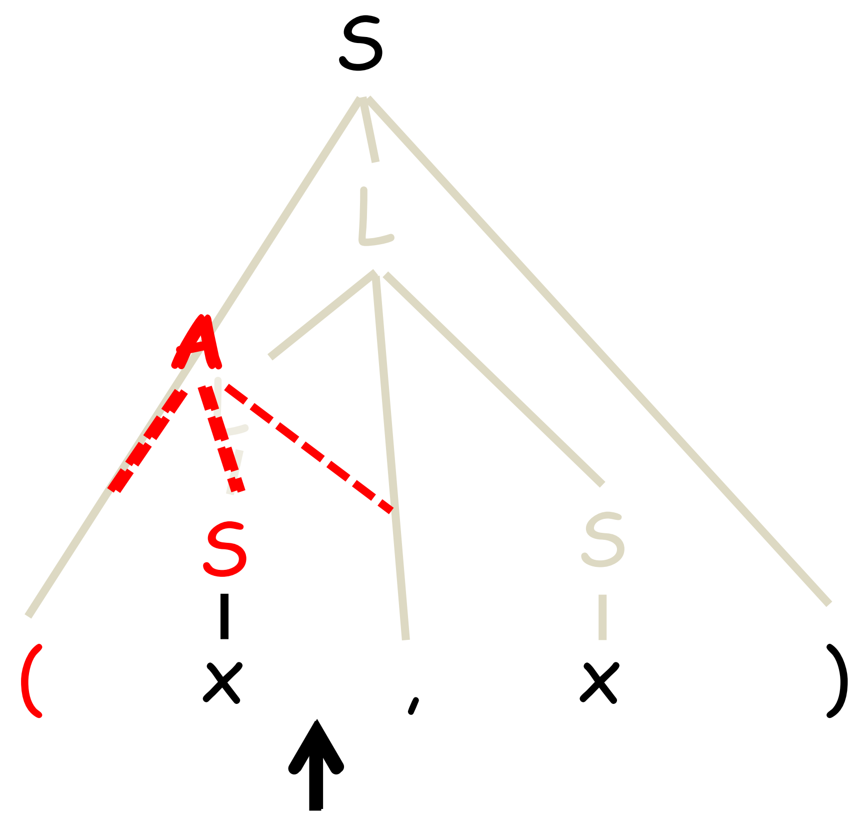
$A \to X_1 \ldots X_k \,\cdot$

$A \to X_1 \ldots X_j \cdot X_{j+1} \ldots X_k$

問題点：　shift/reduce の選択は？

S'→S$　S→(L)　S→x　L→S　L→L,S

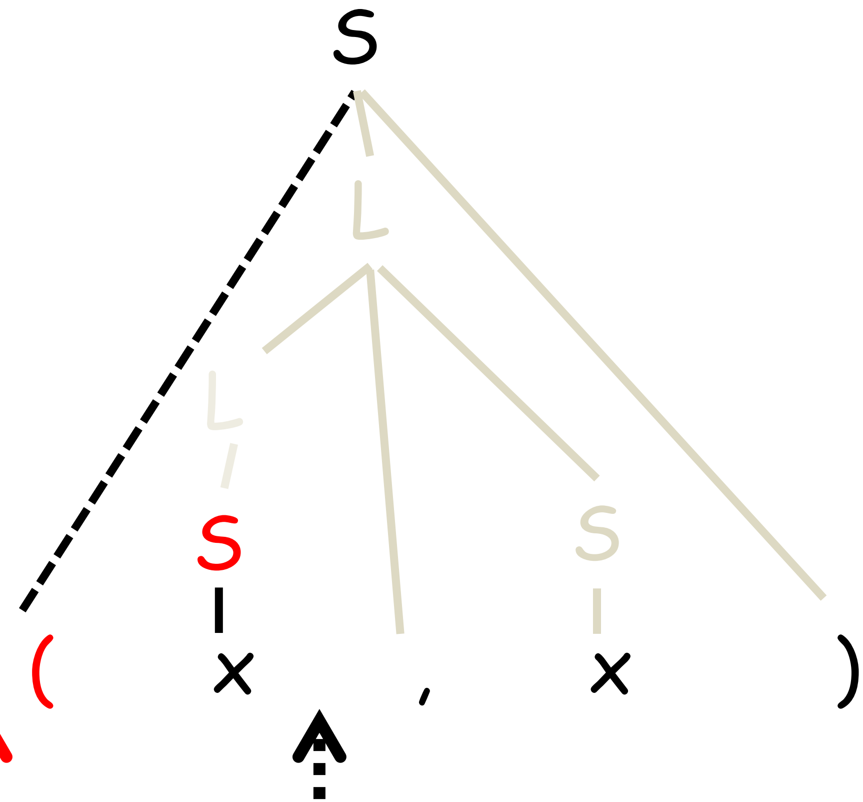
スタック	入力
	(x,x)$
(	x,x)$
(x	,x)$
(S	,x)$



問題点： shift/reduce の選択は？

S'→S$　S→(L)　S→x　L→S　L→L,S

スタック	入力
	(x,x)$
(	x,x)$
(x	,x)$
(S	,x)$



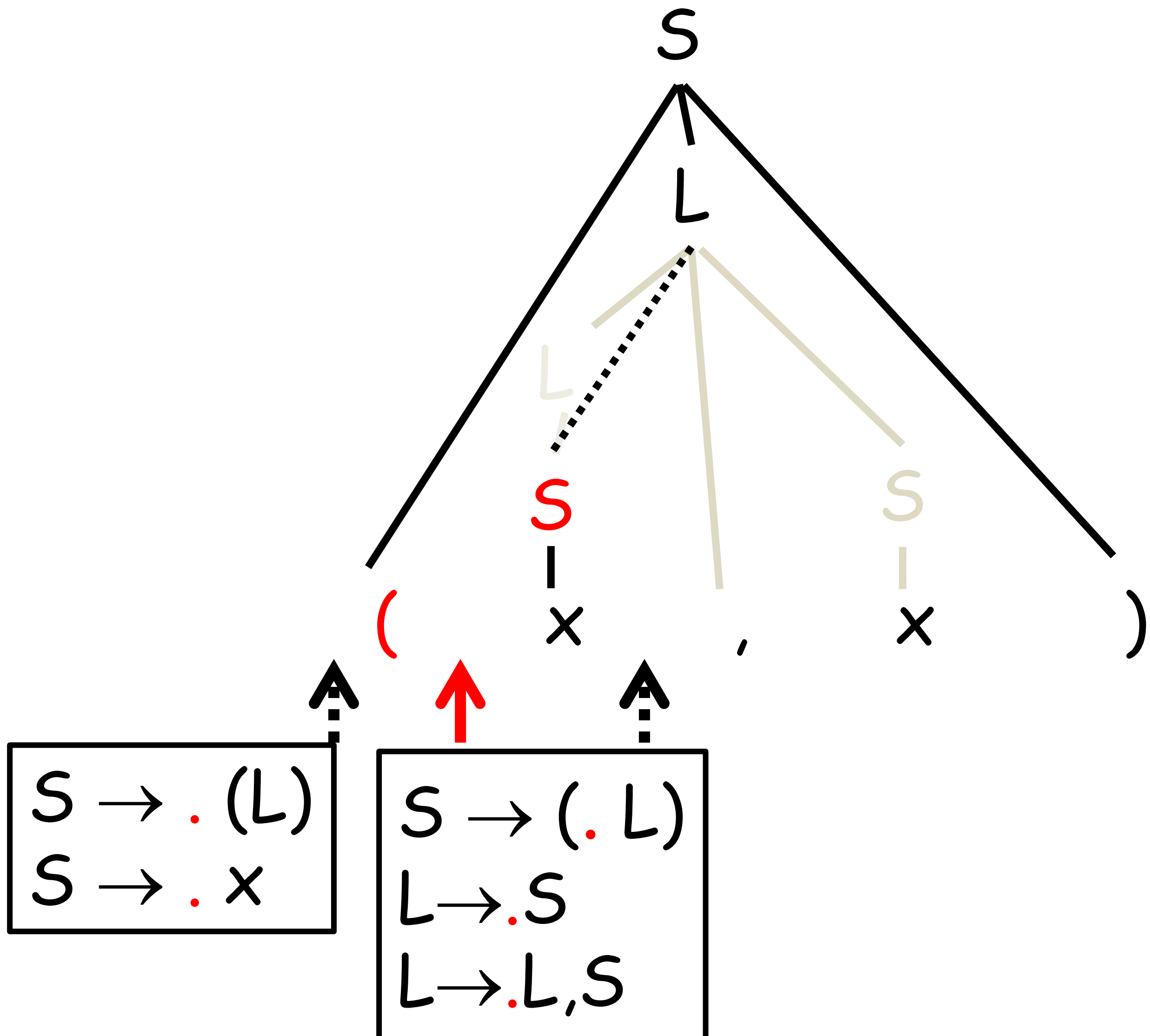
$S \to .\ (L)$

$S \to .\ x$

問題点： shift/reduce の選択は？

S'→S$　S→(L)　S→x　L→S　L→L,S

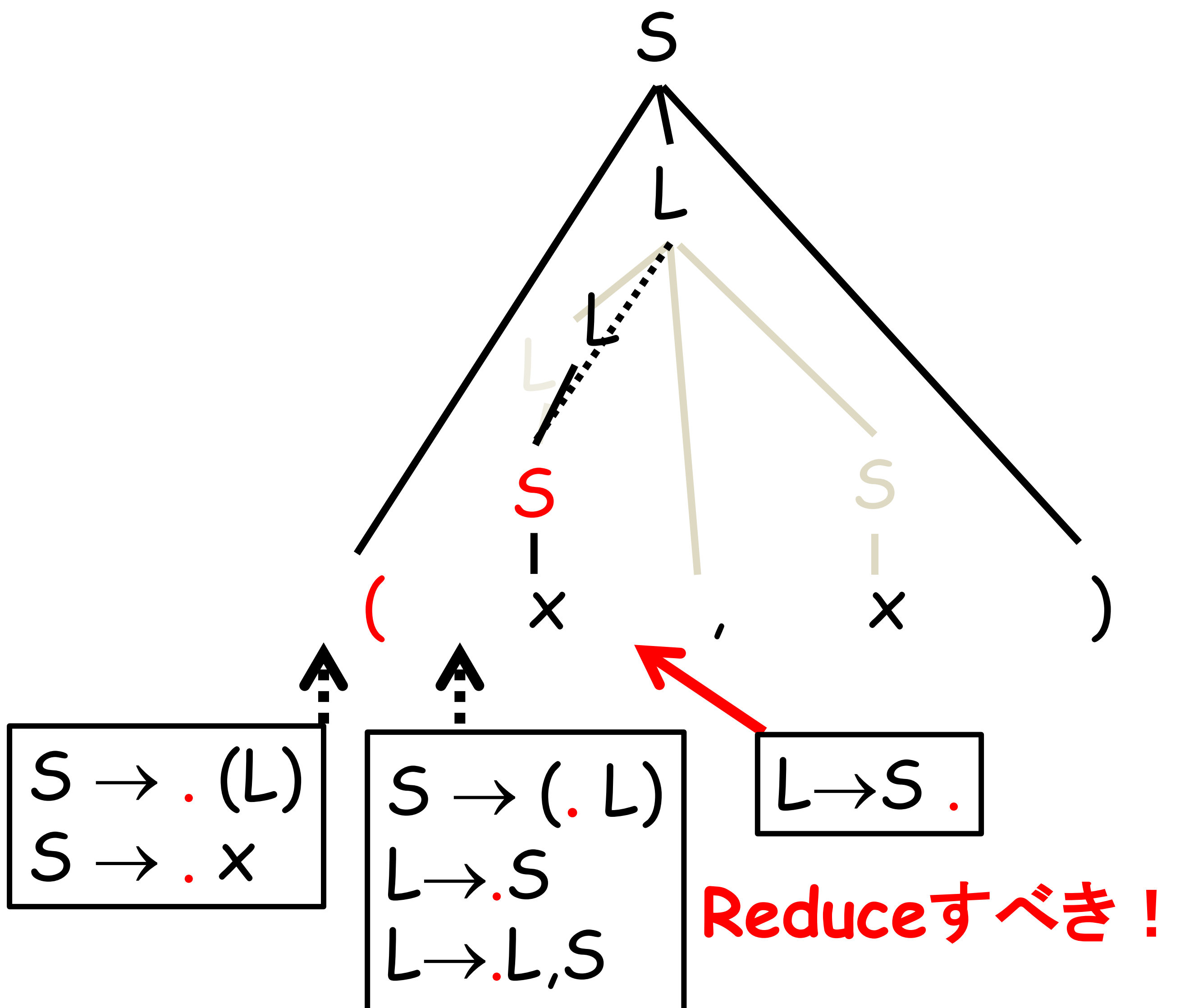
スタック	入力
	(x,x)$
(	x,x)$
(x	,x)$
(S	,x)$



問題点： shift/reduce の選択は？

S'→S$　S→(L)　S→x　L→S　L→L,S

スタック	入力
	(x,x)$
(	x,x)$
(x	,x)$
(S	,x)$



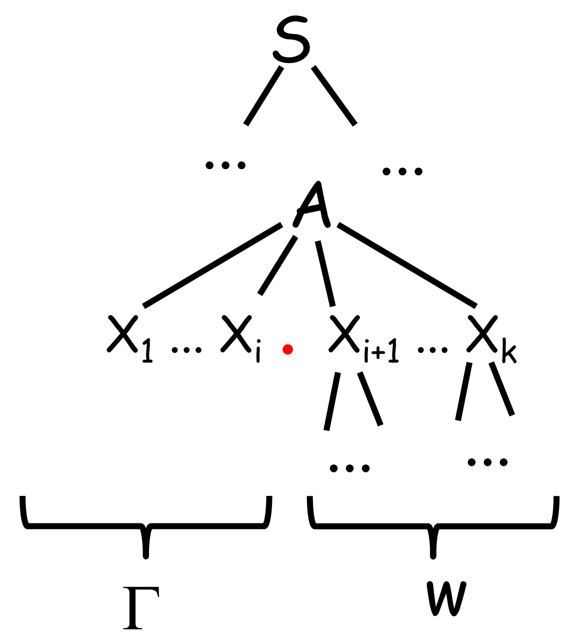
LR(0)

- 各状態(Γ, w)でΓのみからreduceすべきか否かを判断
- 各時点で「現在読んでいる場所がどの規則のどの部分か」を表す情報(「アイテム」)を保持

  $A \to X_1 \ldots X_i \,.\, X_{i+1} \ldots X_k$

  「Aを認識しようとして$X_1 \ldots X_i$まで読み、その後に$X_{i+1} \ldots X_k$から生成される語が来ることを期待している状態」

  (現在のスタックが$\Gamma X_1 \ldots X_i$の形であり、この先$X_{i+1} \ldots X_k$をプッシュした後にAにreduceすべき状態)



- 現時点の「アイテム」が

  $A \to X_1 \ldots X_k \,.$

  のときのみ$A \to X_1 \ldots X_k$に従いreduce

アイテム集合とクロージャ操作

- 各時点の「アイテム」は一つに定まらない

e.g. アイテムが$A \rightarrow B.\ C$ のとき、Cの先頭を処理しているともみなせるので$C \rightarrow .\ \gamma$も考慮の必要あり

⇒各時点でアイテムの「集合」を保持

- クロージャ(closure)操作

closure(q) =

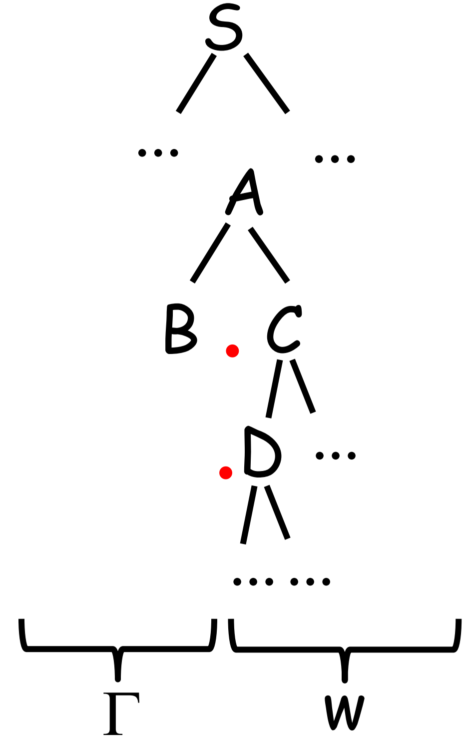
q を含み、

「$A \rightarrow \alpha\ .\ C \beta \in q'$ かつ

$C \rightarrow \gamma \in G$ ならば

$C \rightarrow .\ \gamma \in q'$」

を満たす最小の集合q'



アウトライン

- LR構文解析
- LR(0)
  - shift/reduceの判断法
  - LR(0)オートマトン
  - LR(0)アルゴリズムの実際
  - 形式言語理論の立場からみたLR(0)の仕組み
- LR(0)の改良版
  - SLR
  - LR(k)
  - LALR

LR(0)オートマトン

- 解析途中の各状態(Γ, w)でΓから現在のアイテム集合を計算するのに使用
- 構成要素:
  - 状態: アイテム集合qでclosure(q)=qを満たすもの
    (「LR(0)状態」と呼ぶ)
  - 初期状態: q_0 = closure($\{S \to . \alpha\}$)
  - 入力アルファベット: 文法の非終端記号および終端記号
  - 遷移規則:
    $$\delta(q, X) = \text{closure}(\{ A \to \alpha X . \beta \mid A \to \alpha . X \beta \in q\})$$
- 各解析状態$(X_1 \ldots X_n, w)$でのLR(0)オートマトンの状態:

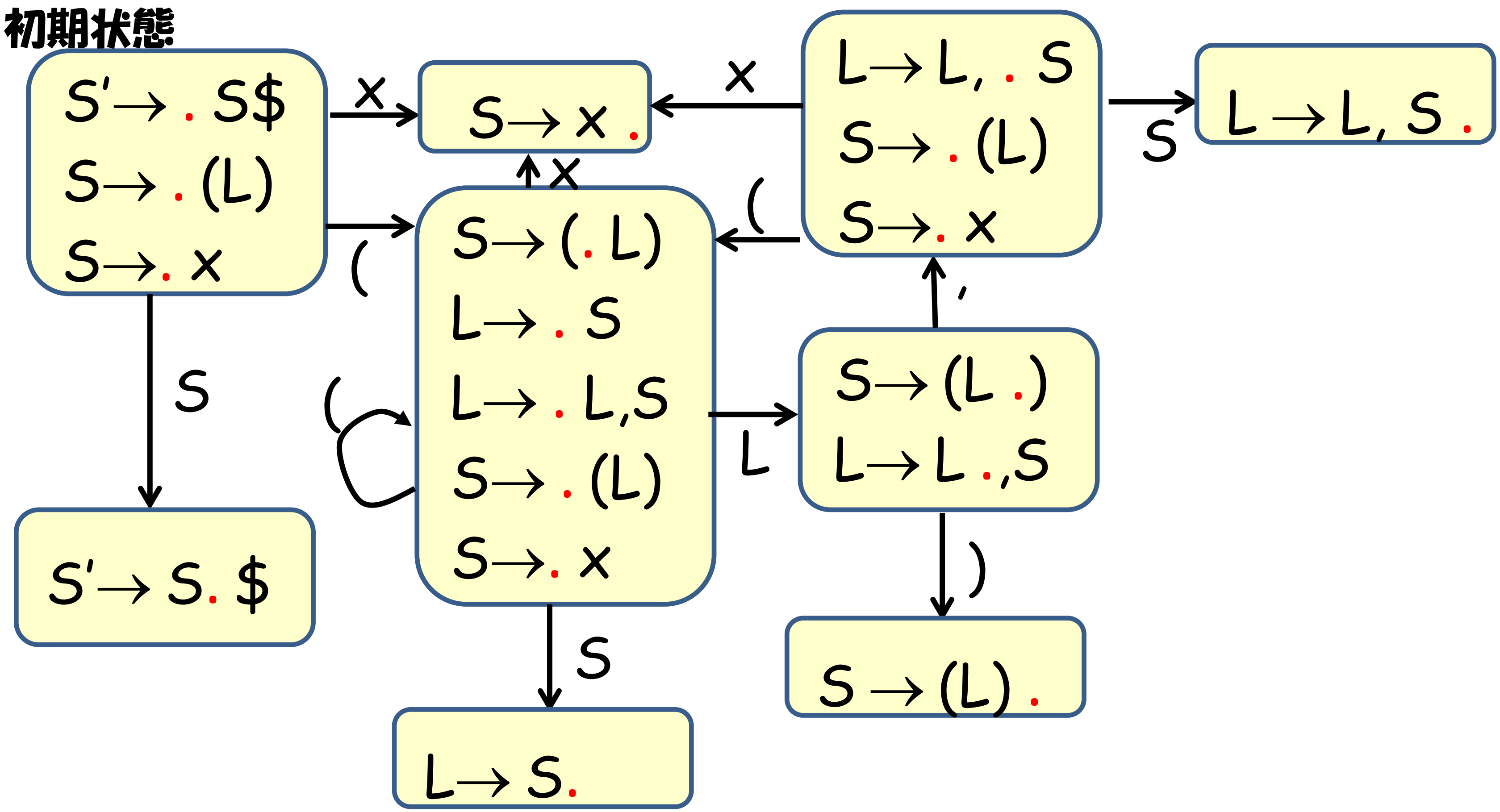
$$q_0 \xrightarrow[X_1]{} q_1 \rightarrow \ \ldots \xrightarrow[X_n]{} q_n$$ を満たすq_n

例：{S'→S$;　S→(L);　S→x; L→S; L→L,S}
のLR(0)オートマトン (教科書 p.61)

初期

黒板で

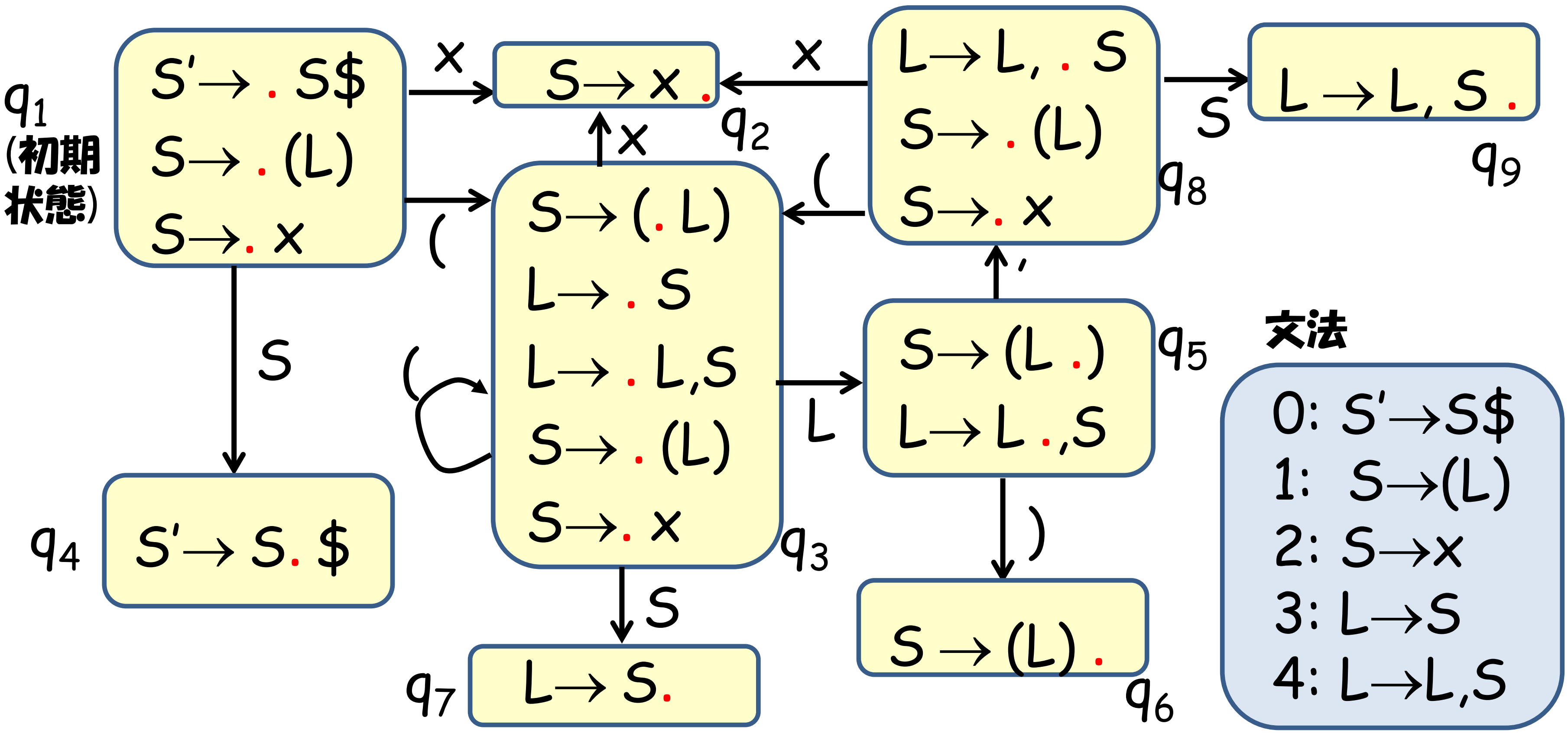
例：{S'→S$;　S→(L);　S→x; L→S; L→L,S} のLR(0)オートマトン (教科書 p.61)



例:LR(0)オートマトンを用いたshift/reduceの選択

解析の状態: スタック: (L,S　　入力バッファ: ,x)
- shift すべきか　reduce すべきか？
- reduce するなら L → S かL → L,S か？

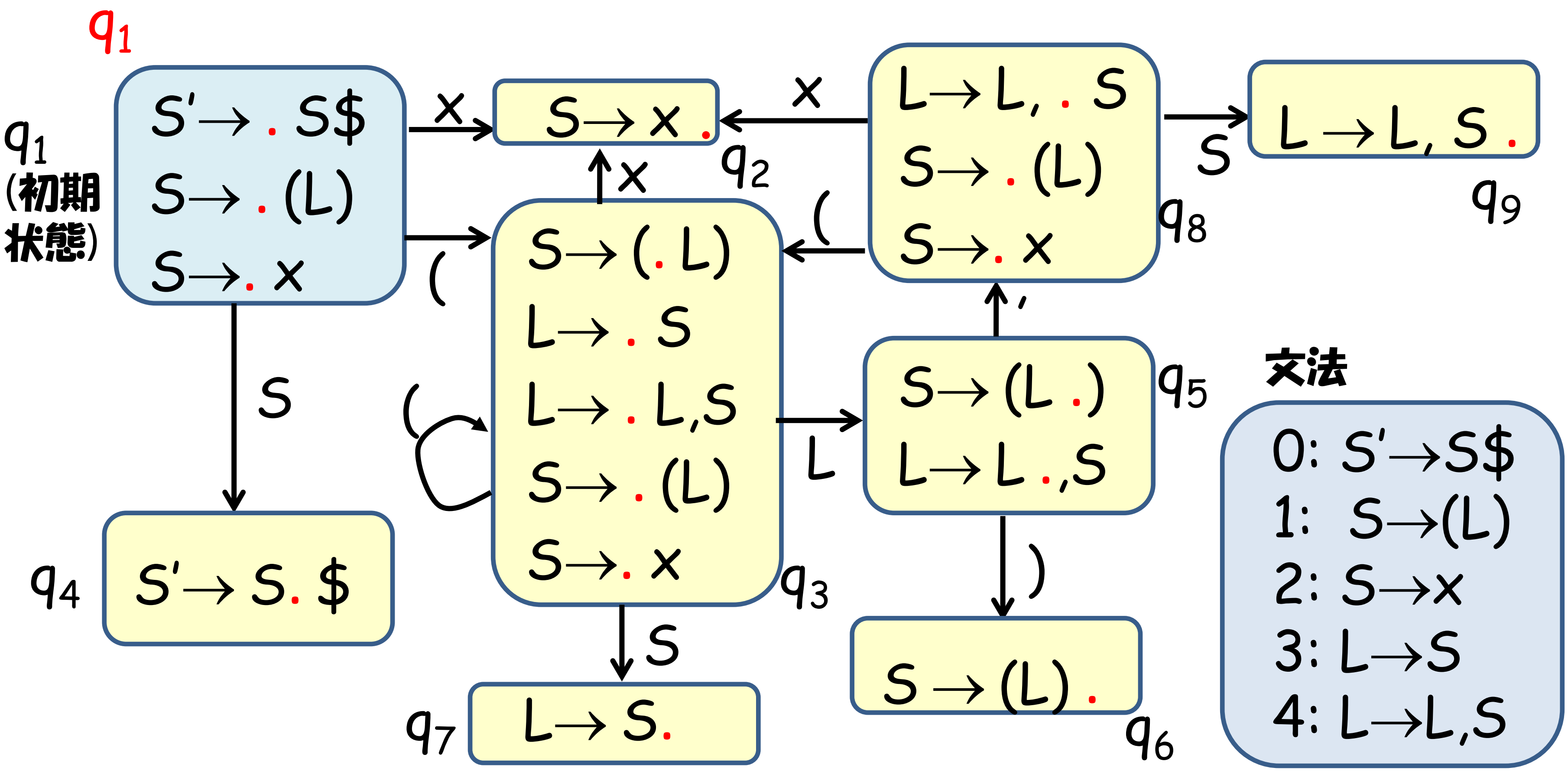
⇒ スタック上の記号列に対してLR(0)オートマトンを実行



例:LR(0)オートマトンを用いたshift/reduceの選択

解析の状態: スタック: (L,S　　入力バッファ: ,x)
- shift すべきか　reduce すべきか？
- reduce するなら L → S かL → L,S か？

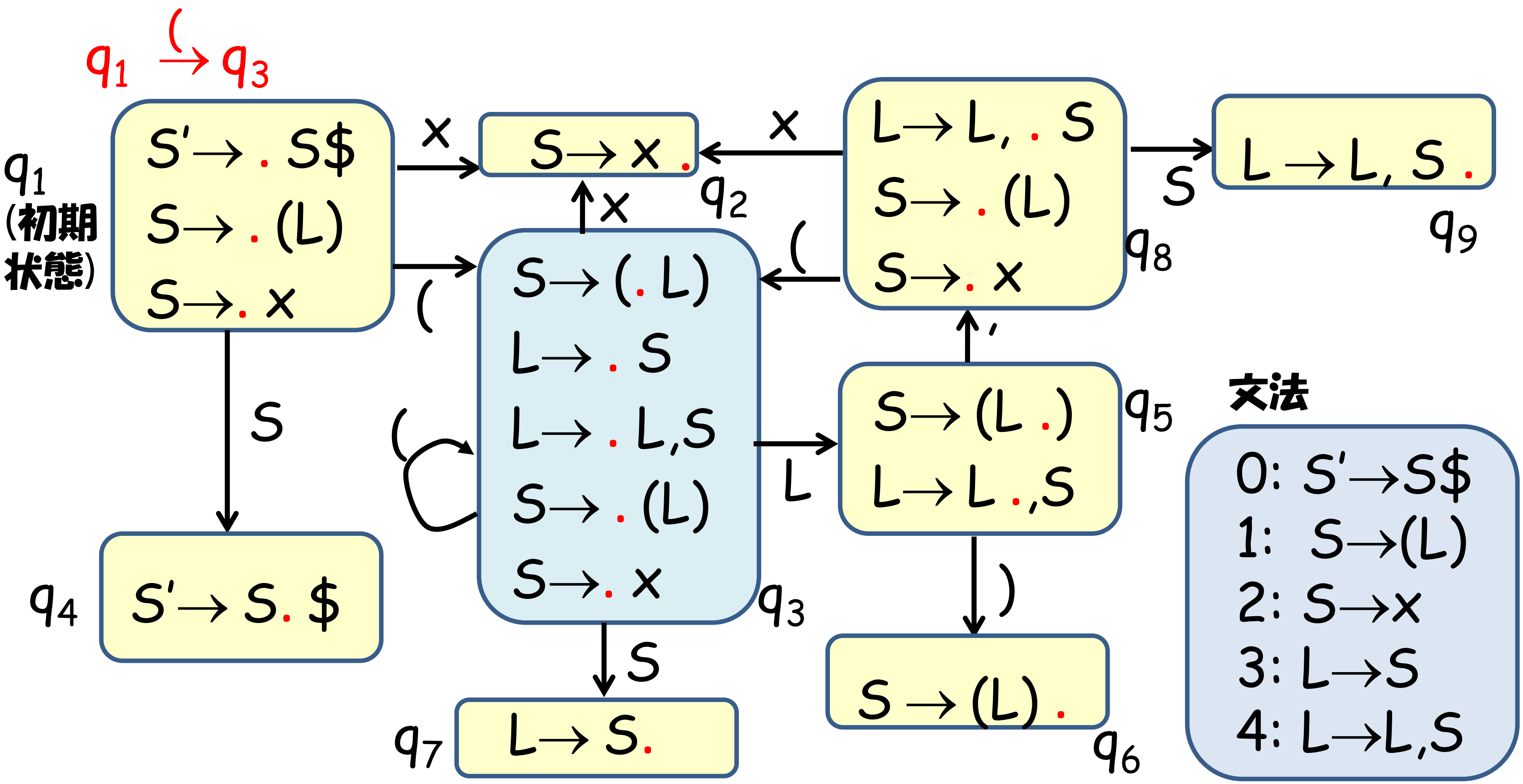
⇒ スタック上の記号列に対してLR(0)オートマトンを実行



例:LR(0)オートマトンを用いたshift/reduceの選択

解析の状態: スタック: (L,S　　入力バッファ: ,x)
- shift すべきか reduce すべきか？
- reduce するなら L→S かL→L,S か？

⇒ スタック上の記号列に対してLR(0)オートマトンを実行

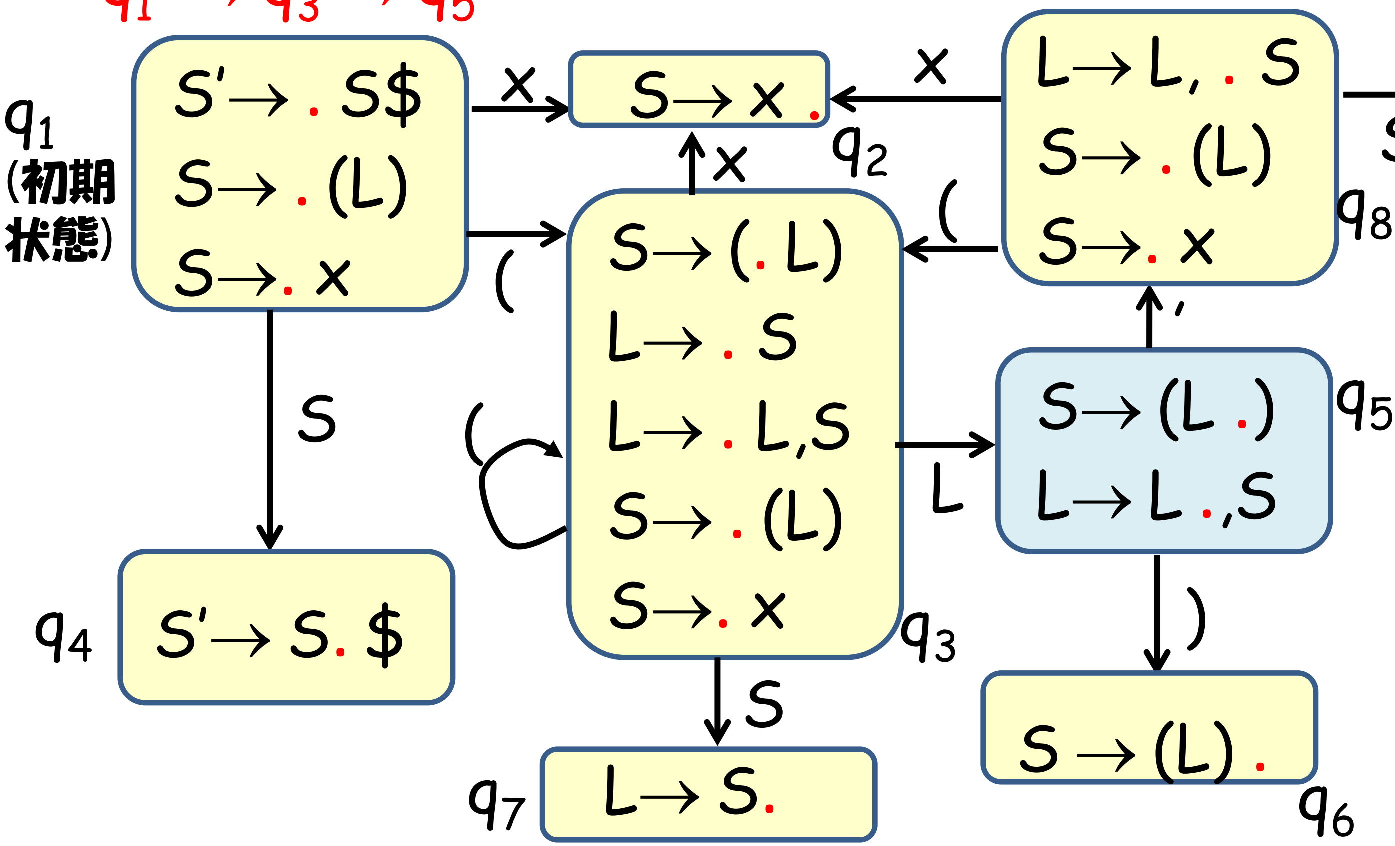


例：LR(0)オートマトンを用いたshift/reduceの選択

解析の状態： スタック： (L,S 　　入力バッファ： ,x)
- shift すべきか reduce すべきか？
- reduce するなら L→S かL→L,S か？

⇒ スタック上の記号列に対してLR(0)オートマトンを実行

$q_1 \xrightarrow{(} q_3 \xrightarrow{L} q_5$



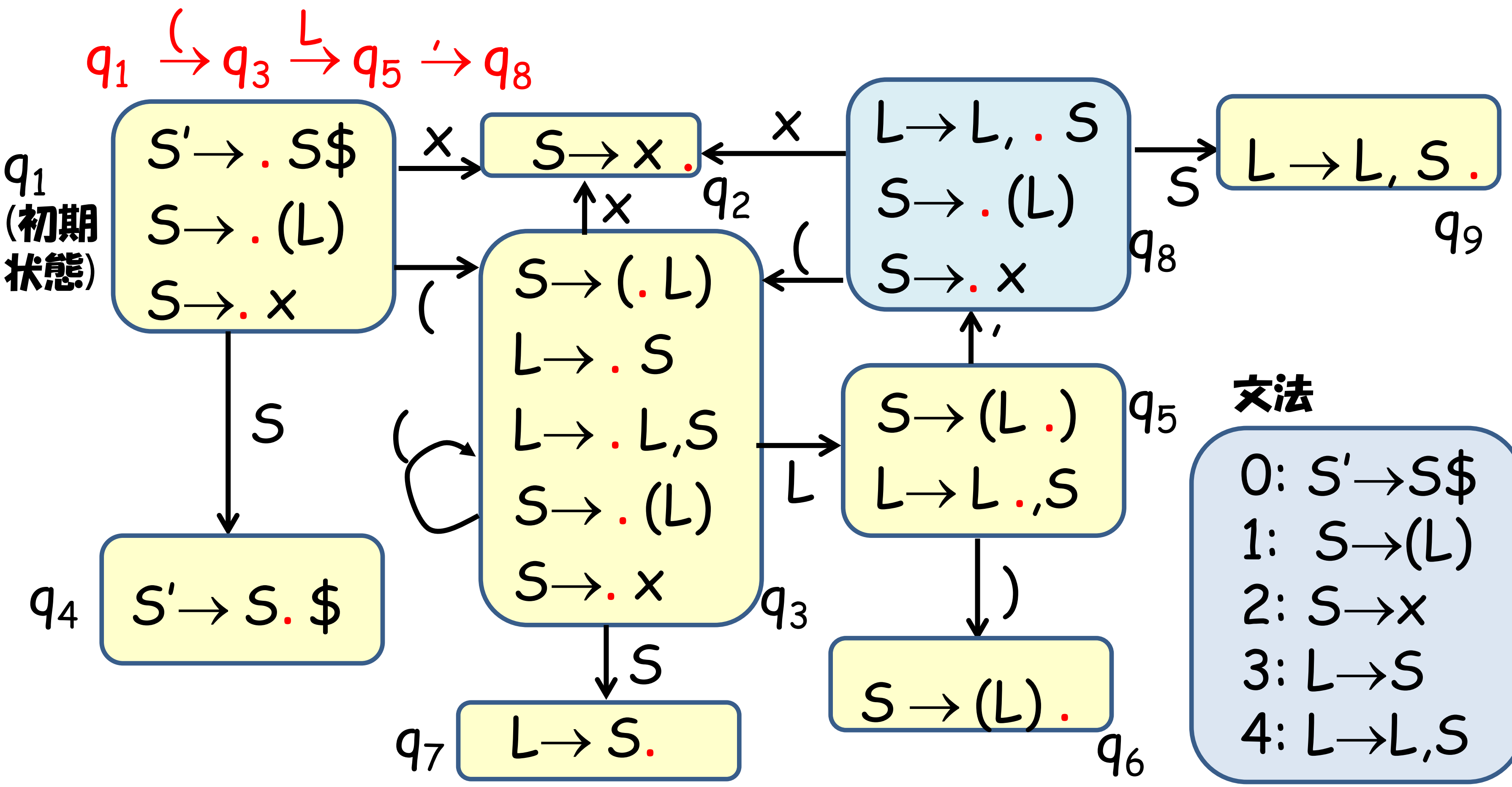
文法

0: S'→S$
1: S→(L)
2: S→x
3: L→S
4: L→L,S

例：LR(0)オートマトンを用いたshift/reduceの選択

解析の状態：スタック：　(L,S　　　入力バッファ：　,x)
- shift すべきか　reduce すべきか？
- reduce するなら L → S かL → L,S か？

⇒ スタック上の記号列に対してLR(0)オートマトンを実行



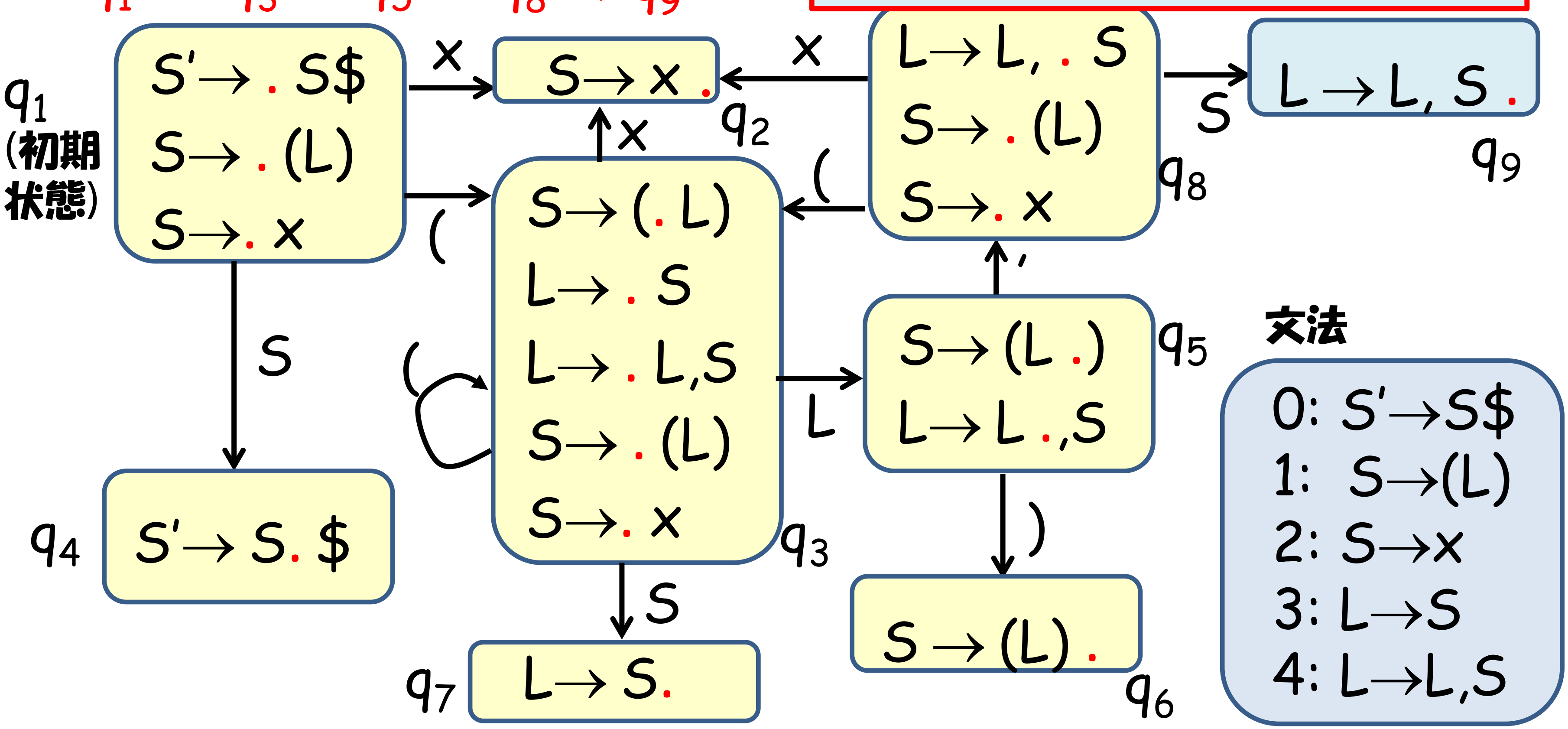
例：LR(0)オートマトンを用いたshift/reduceの選択

解析の状態：スタック：　(L,S　　　入力バッファ：,x)
- shift すべきか　reduce すべきか？
- reduce するなら L→S かL→L,S か？

⇒ スタック上の記号列に対してLR(0)オートマトンを実行

$q_1 \xrightarrow{(} q_3 \xrightarrow{L} q_5 \xrightarrow{,} q_8 \xrightarrow{S} q_9$

L→L, S によってreduce!



アウトライン

- LR構文解析
- LR(0)
  - shift/reduceの判断法
  - LR(0)オートマトン
  - LR(0)アルゴリズムの実際
  - 形式言語理論の立場からみたLR(0)の仕組み
- LR(0)の改良版
  - SLR
  - LR(k)
  - LALR

LR(0)アルゴリズムの実際

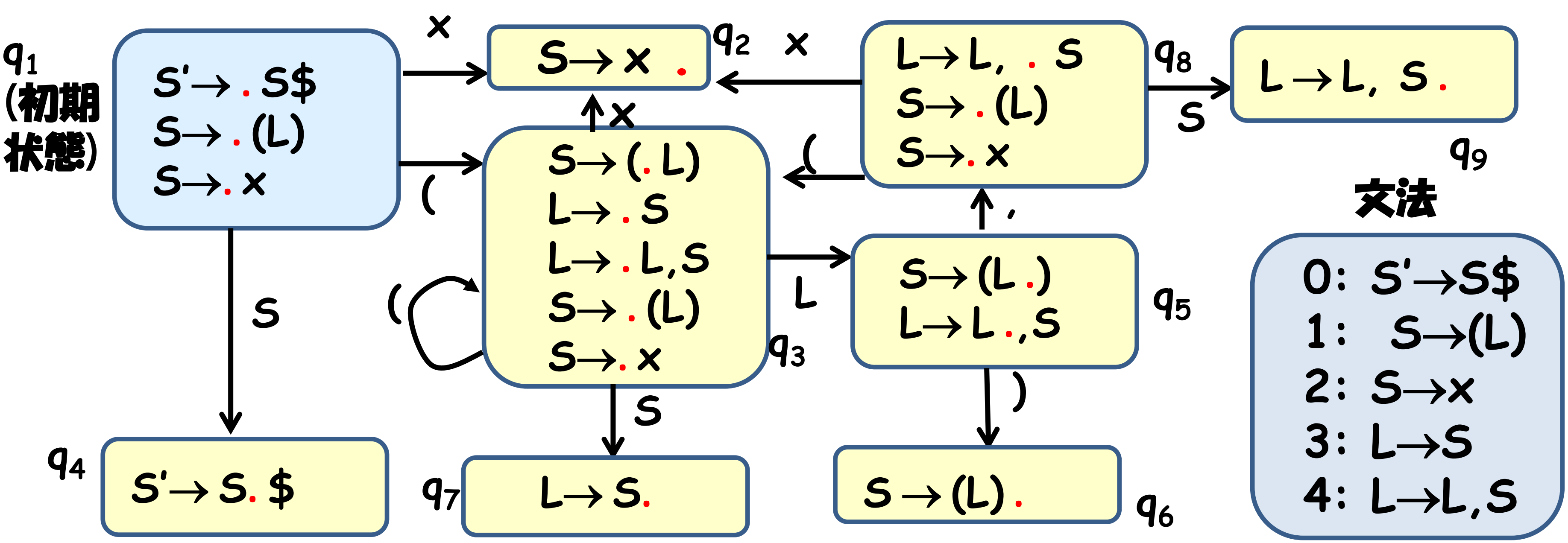
- スタックに各時点のLR(0)状態を追加
  (reduceのたびにLR(0)状態を最初から計算するのを回避)

  $(q_0\ X_1\ q_1\ \ldots.\ q_{n-1}\ X_n\ q_n\ ,\ \ w)$

  q_j: LR(0)オートマトンに$X_1 \ldots X_j$を入力したあとの状態
- 初期状態 $(q_0,\ w\$)$から以下の操作(アクション)を繰り返し実行
  - shift: $(q_0 X_1\ \ldots\ X_n q_n,\ aw) \to (q_0 X_1\ \ldots\ X_n q_n a q_{n+1},\ aw)$
    if $\delta(q_n, a) = q_{n+1}$ ($\delta(q_n,a)$が未定義なら構文解析エラー)
  - reduce:
    $(q_0 X_1 \ldots X_r\ q_r X_{r+1}\ \ldots X_n q_n\ ,\ w) \to (q_0 X_1 \ldots X_r q_r A q',\ \ w)$
    if $A \to X_{r+1}\ \ldots\ X_n\ .\ \in q_n$ and $\delta(q_r, A) = q'$
  - accept: $(q_0\, \alpha\ q,\ \ \$)$ if $S \to \alpha.\$ \in q$
- 各状態での操作(shift/reduce/accept)をあらかじめ表
  (LR(0)構文解析表:後述)で管理

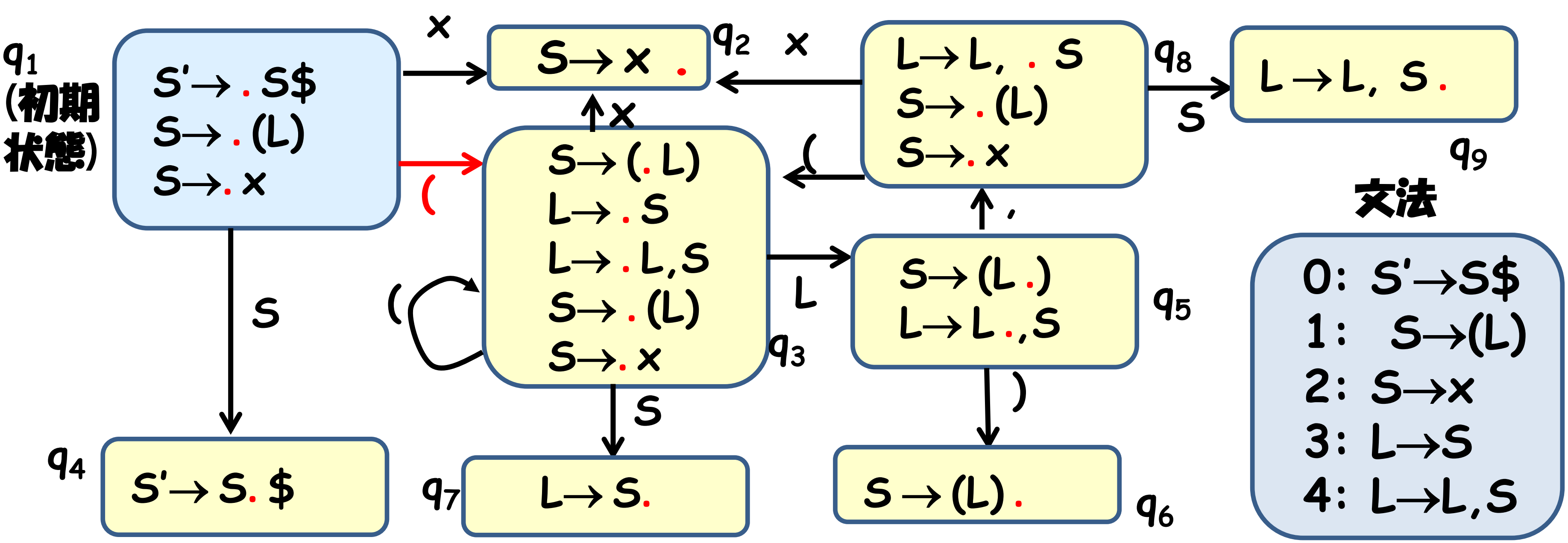
解析例:

スタック	入力	アクション
q_1	(x,x)$	



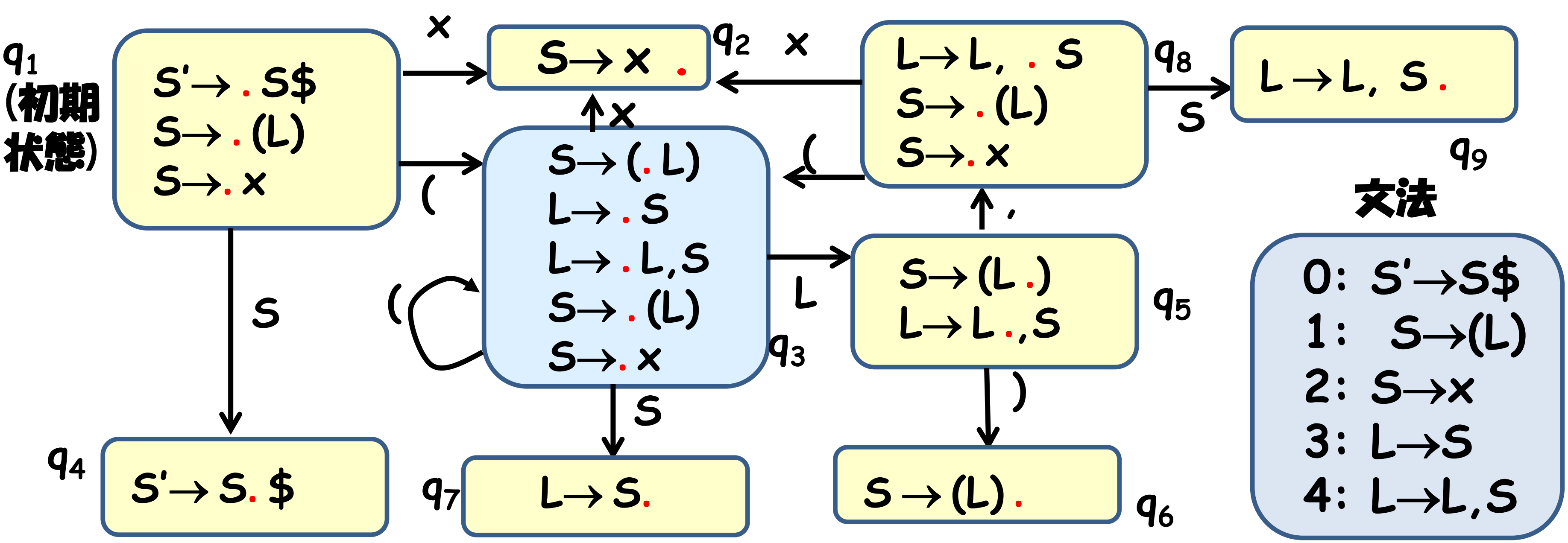
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift



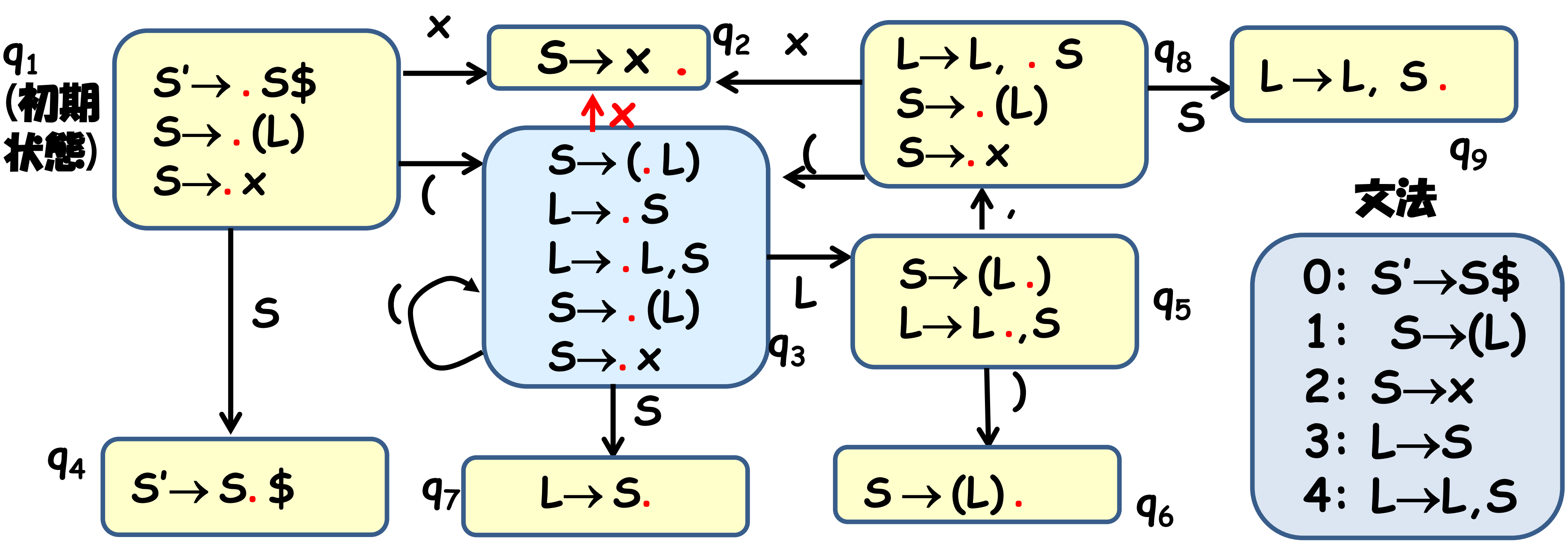
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	



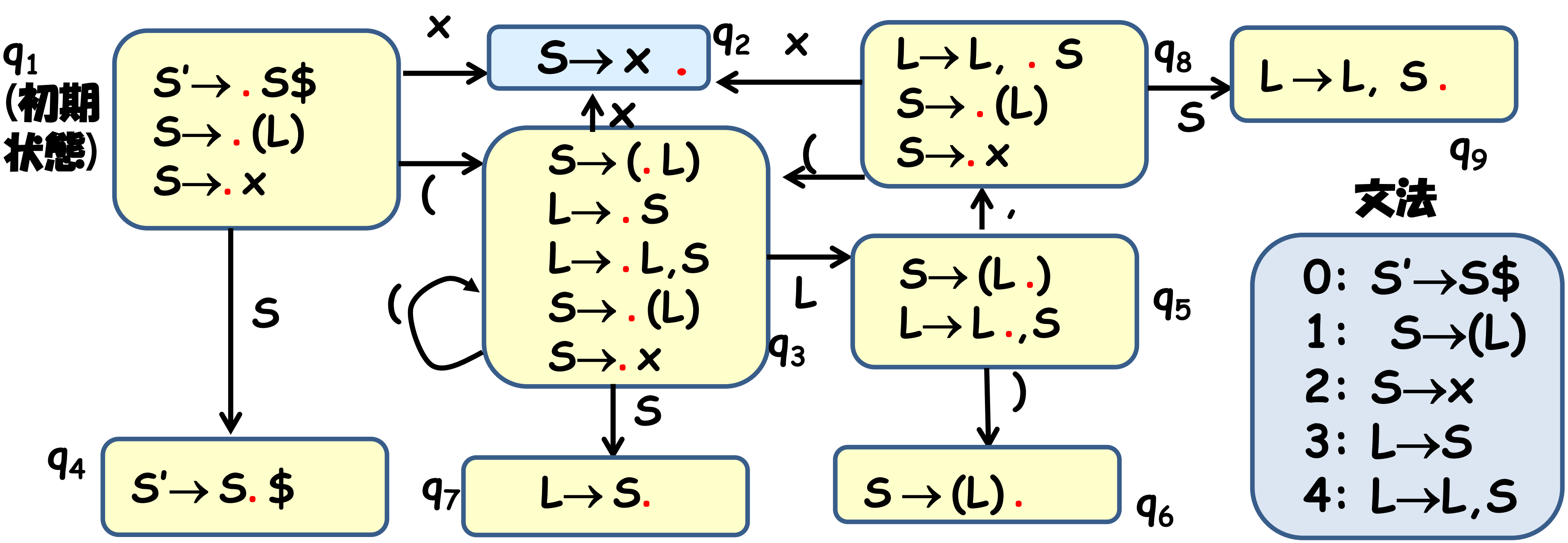
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift



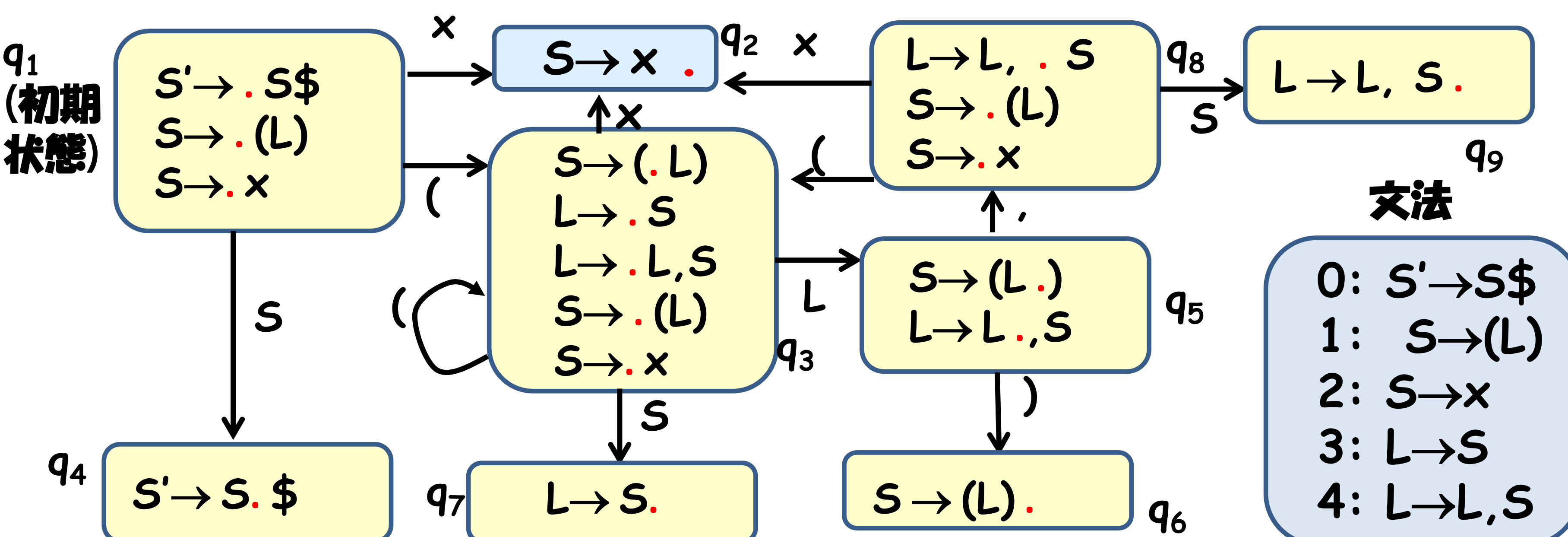
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	



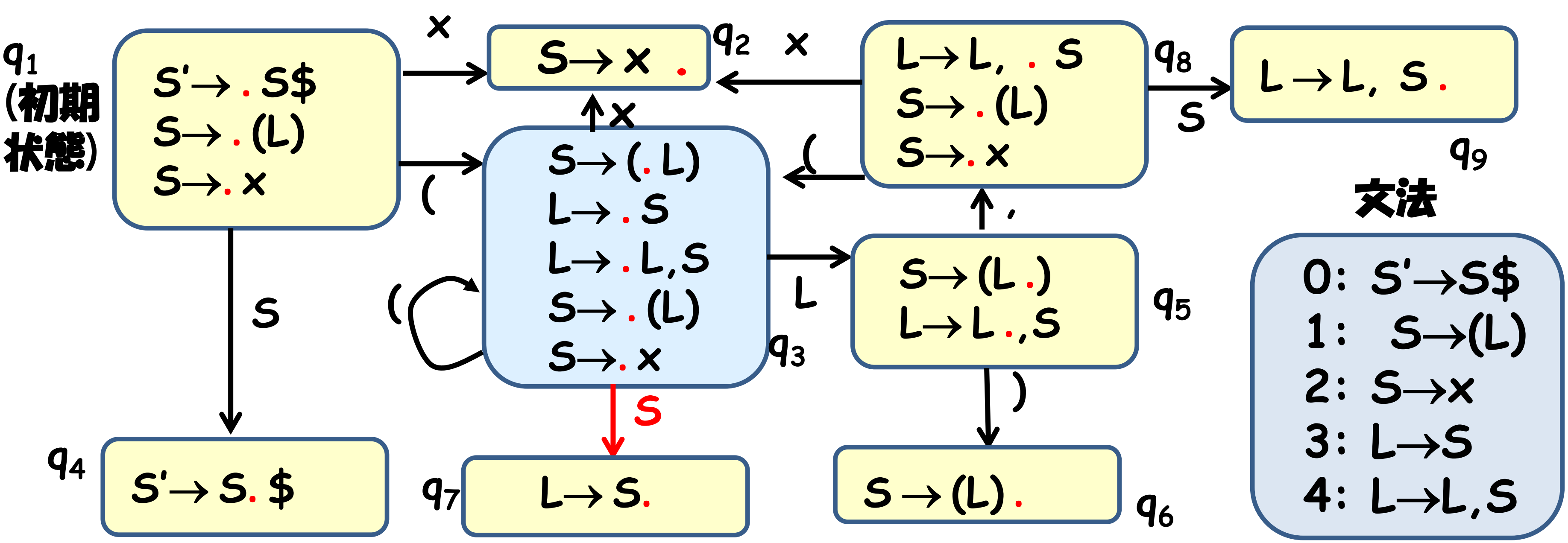
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2



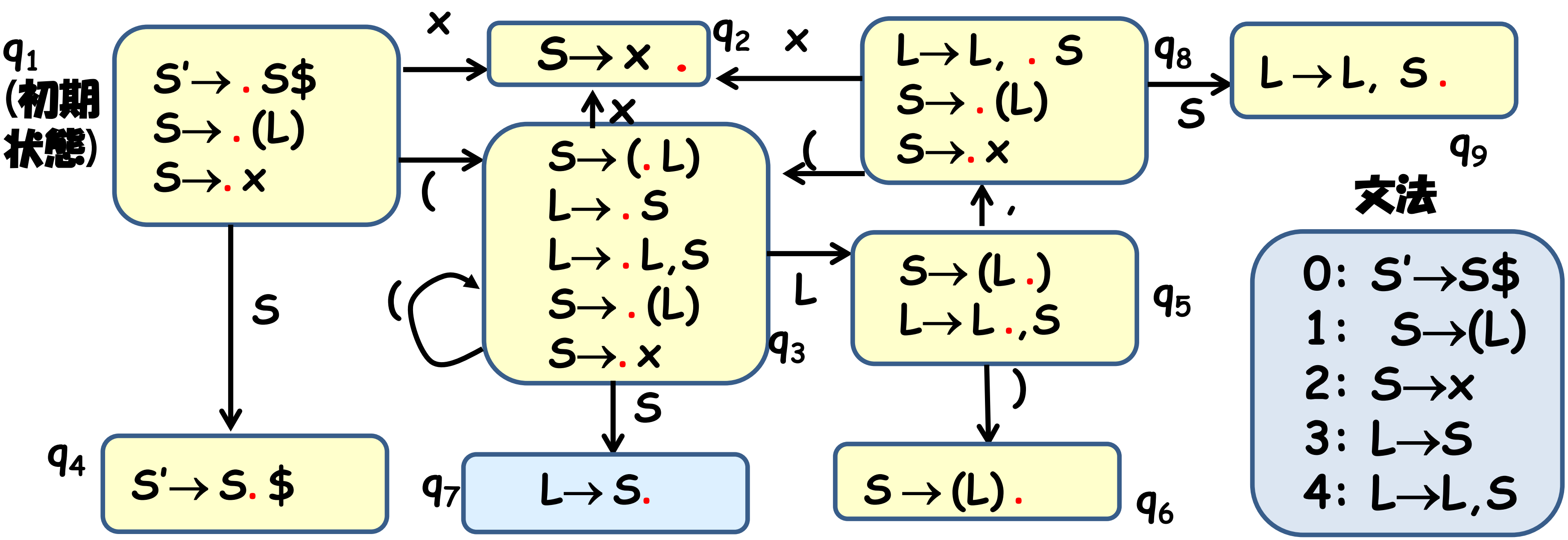
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2



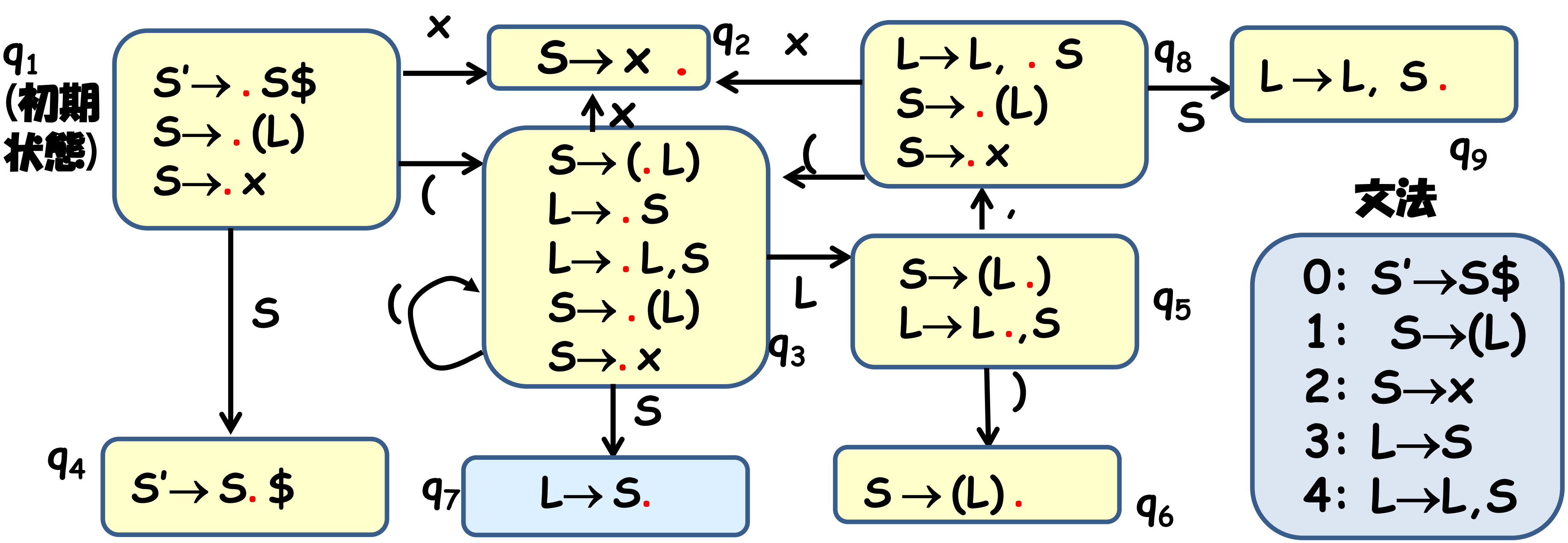
解析例:

スタック	入力	アクション
q_1	(x,x)\$	shift
$q_1(q_3$	x,x)\$	shift
$q_1(q_3 x q_2$	,x)\$	reduce by 2
$q_1(q_3 S q_7$	,x)\$	



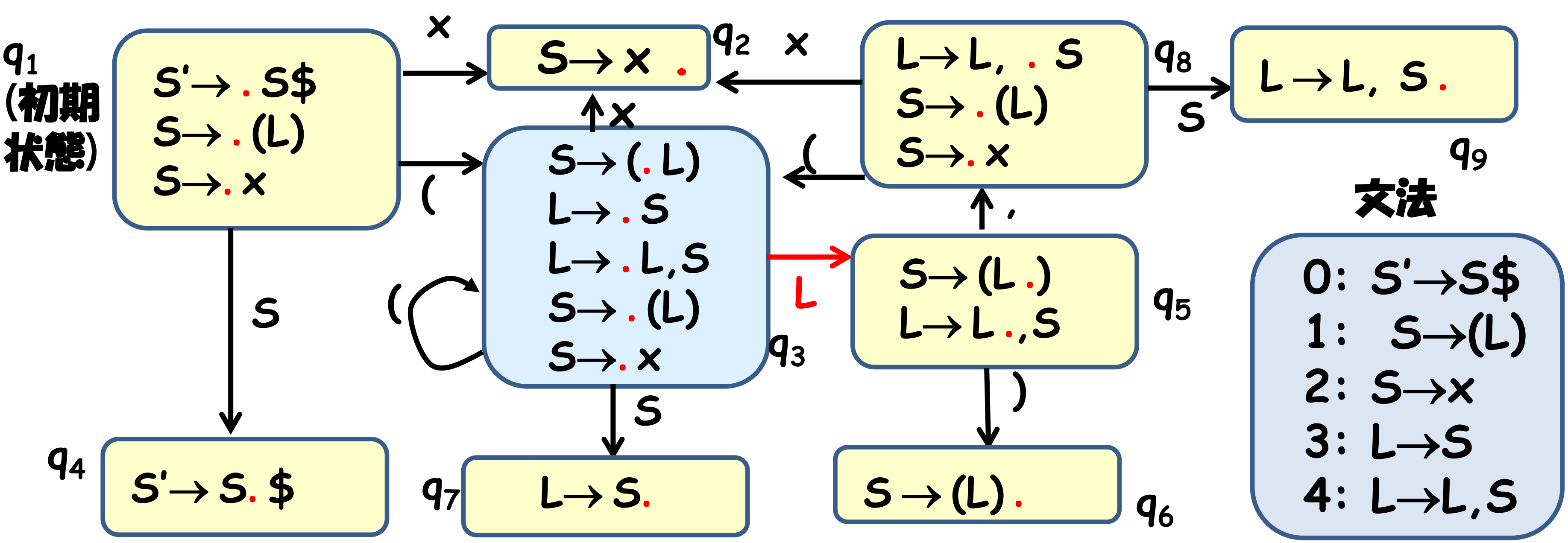
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3



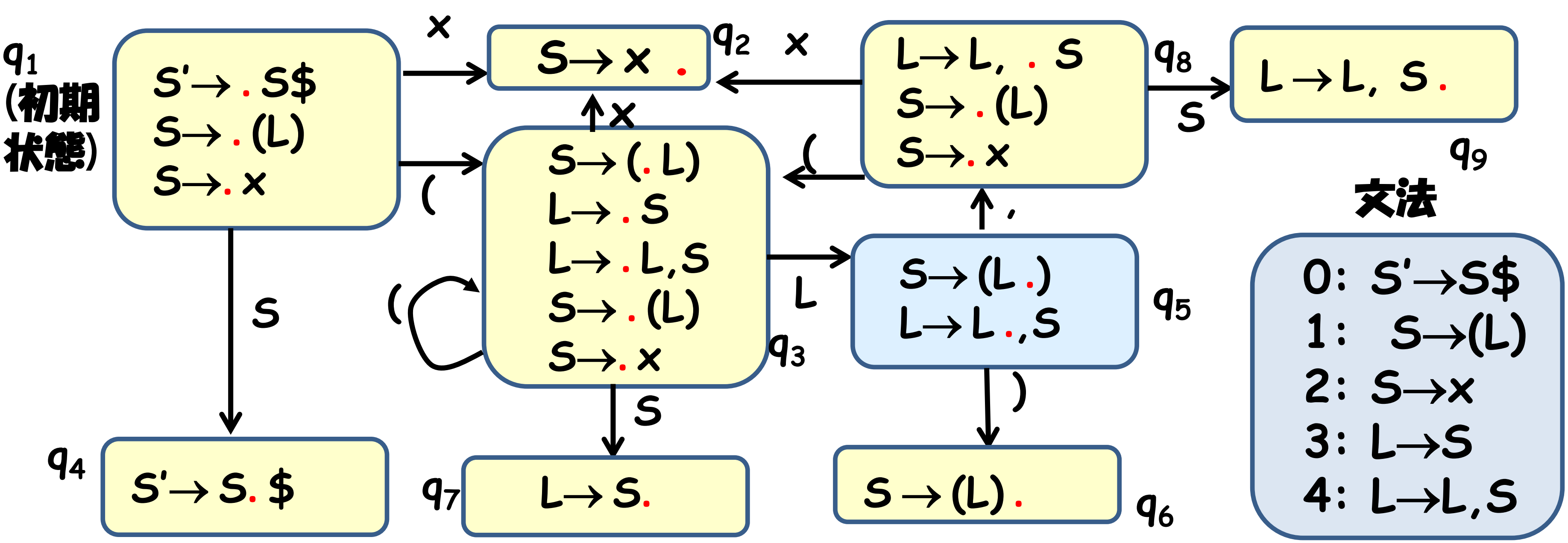
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3



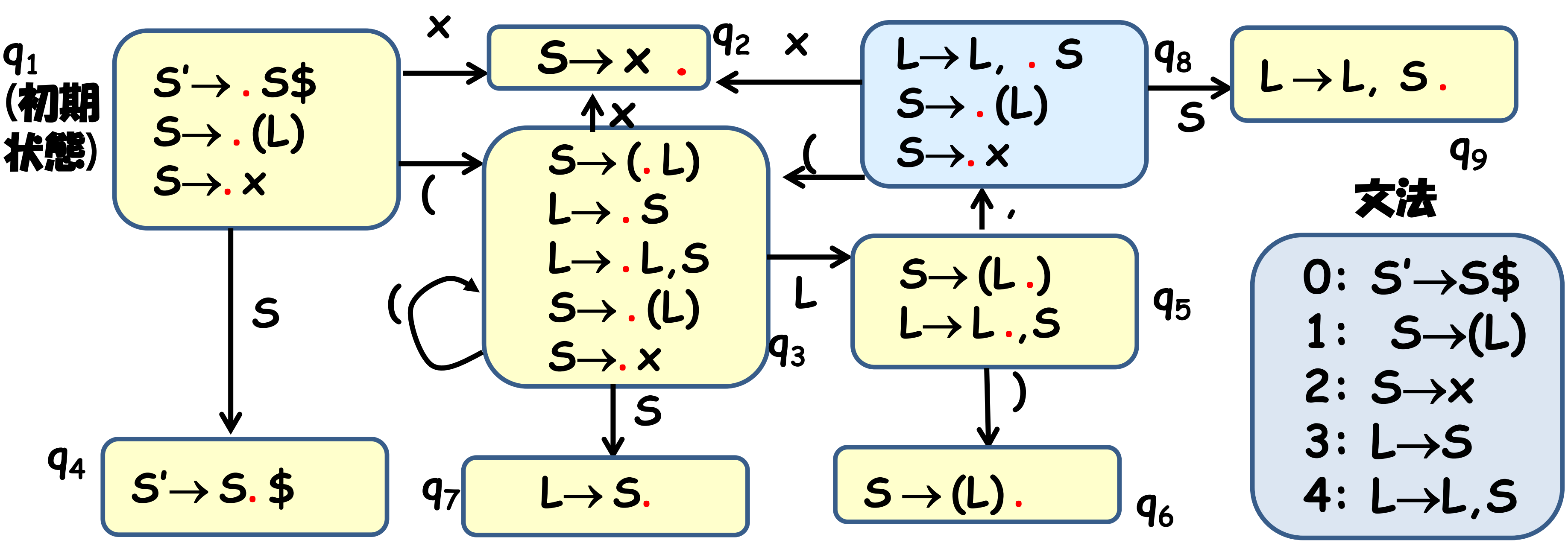
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3
$q_1(q_3 L q_5$	,x)$	shift



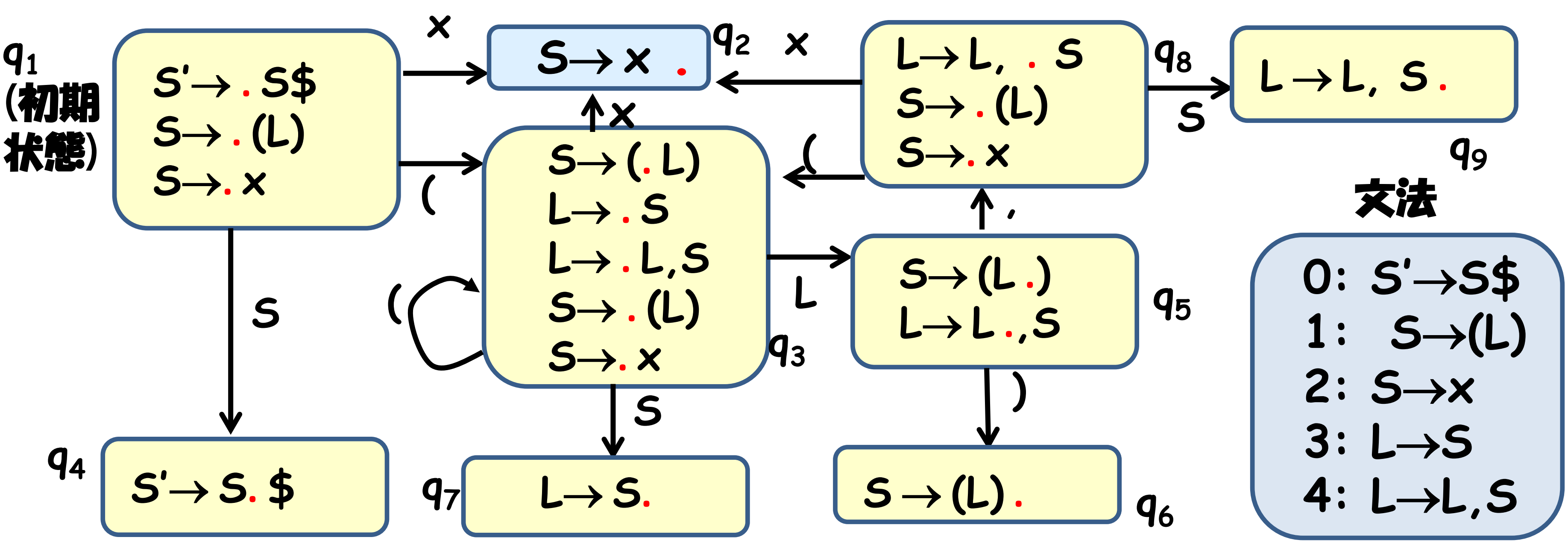
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3xq_2$	,x)$	reduce by 2
$q_1(q_3Sq_7$	,x)$	reduce by 3
$q_1(q_3Lq_5$	,x)$	shift
$q_1(q_3Lq_5,q_8$	x)$	shift



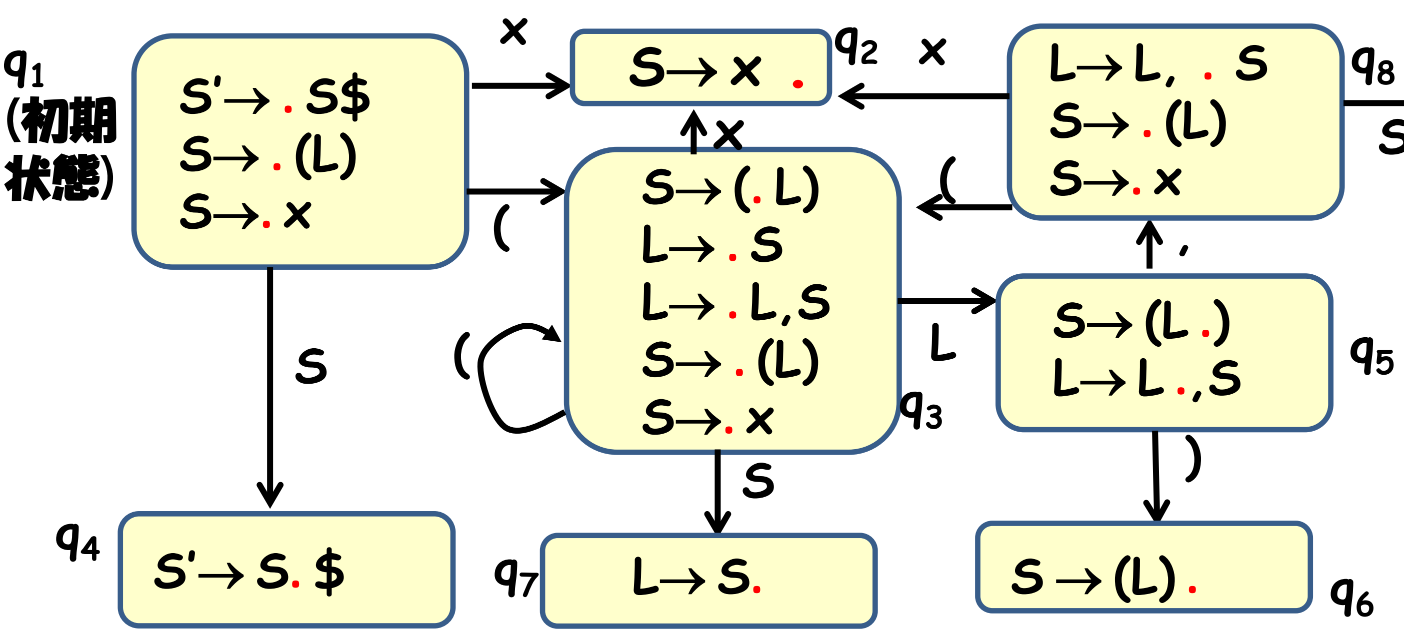
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3
$q_1(q_3 L q_5$	,x)$	shift
$q_1(q_3 L q_5 , q_8$	x)$	shift
$q_1(q_3 L q_5 , q_8 x q_2$	)$	reduce by 2



解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3xq_2$	,x)$	reduce by 2
$q_1(q_3Sq_7$	,x)$	reduce by 3
$q_1(q_3Lq_5$	,x)$	shift
$q_1(q_3Lq_5,q_8$	x)$	shift
$q_1(q_3Lq_5,q_8xq_2$	)$	reduce by 2
$q_1(q_3Lq_5,q_8Sq_9$	)$	reduce by 4

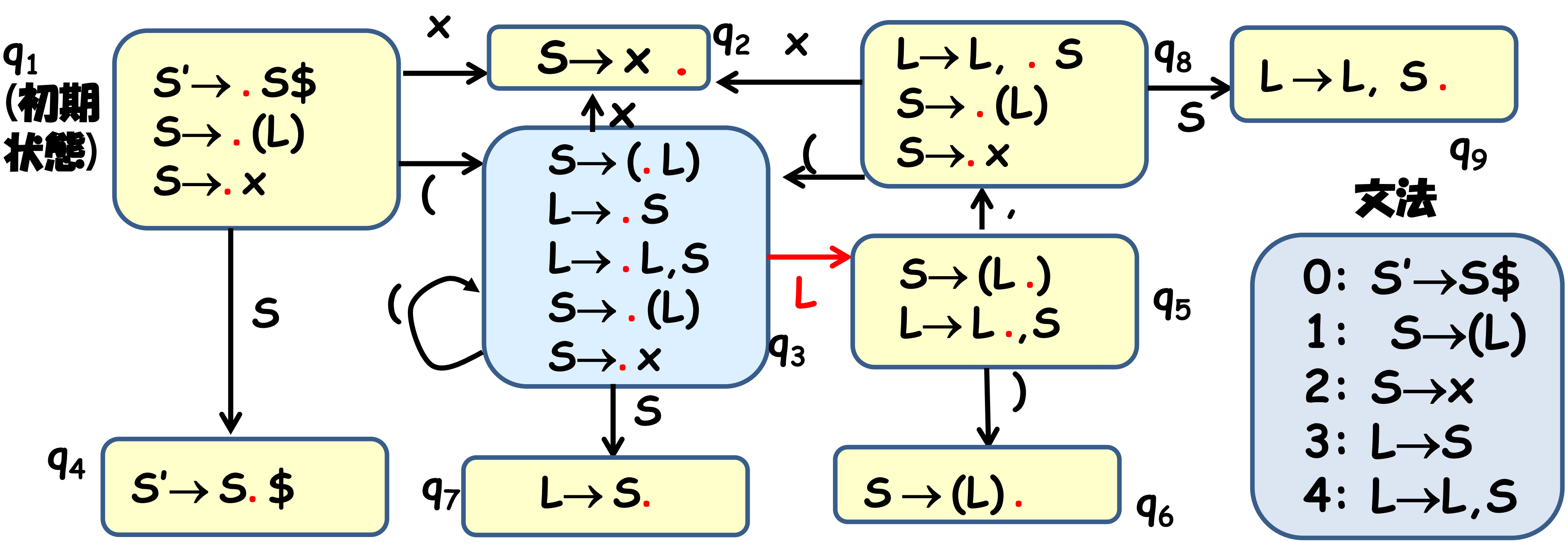


文法

0: S'→S$
1: S→(L)
2: S→x
3: L→S
4: L→L,S

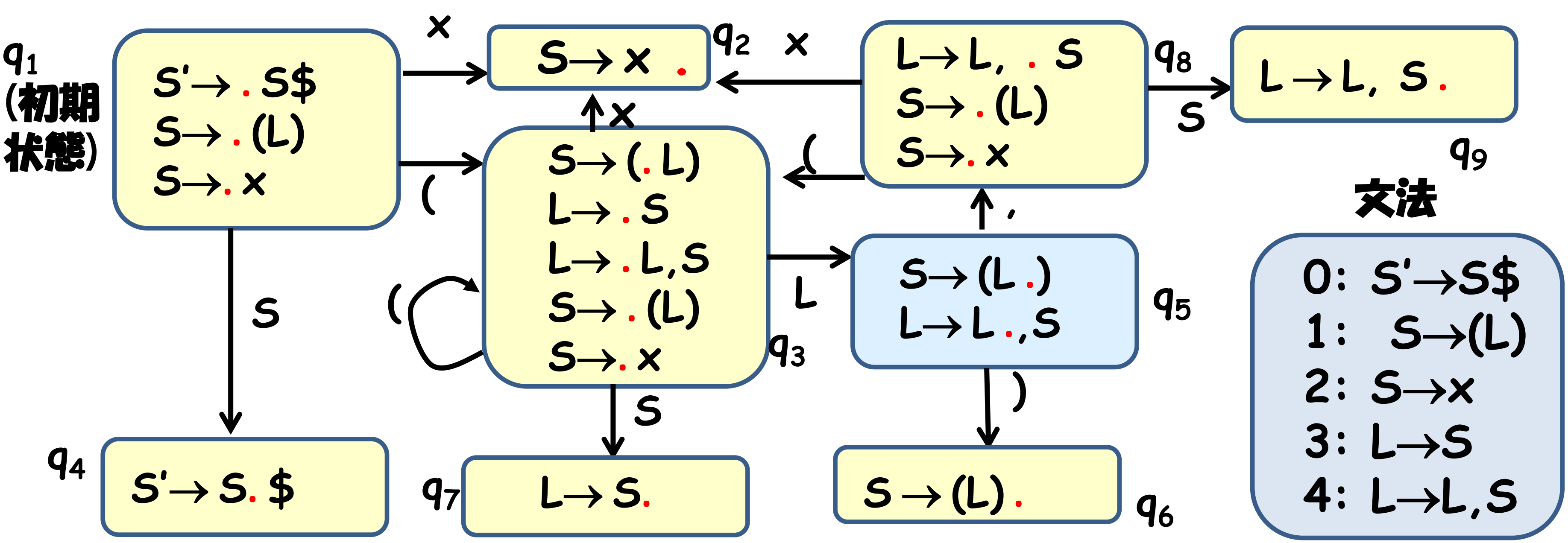
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3
$q_1(q_3 L q_5$	,x)$	shift
$q_1(q_3 L q_5 , q_8$	x)$	shift
$q_1(q_3 L q_5 , q_8 x q_2$	)$	reduce by 2
$q_1(q_3 L q_5 , q_8 S q_9$	)$	reduce by 4



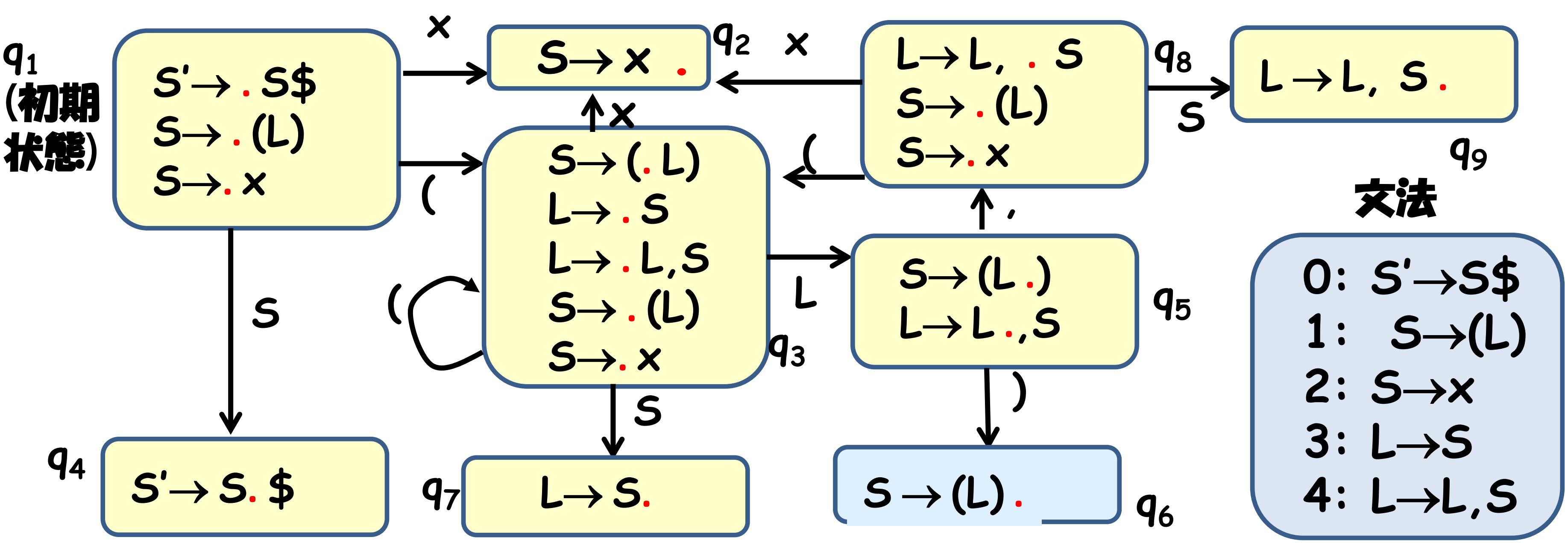
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3
$q_1(q_3 L q_5$	,x)$	shift
$q_1(q_3 L q_5 , q_8$	x)$	shift
$q_1(q_3 L q_5 , q_8 x q_2$	)$	reduce by 2
$q_1(q_3 L q_5 , q_8 S q_9$	)$	reduce by 4
$q_1(q_3 L q_5$	)$	shift



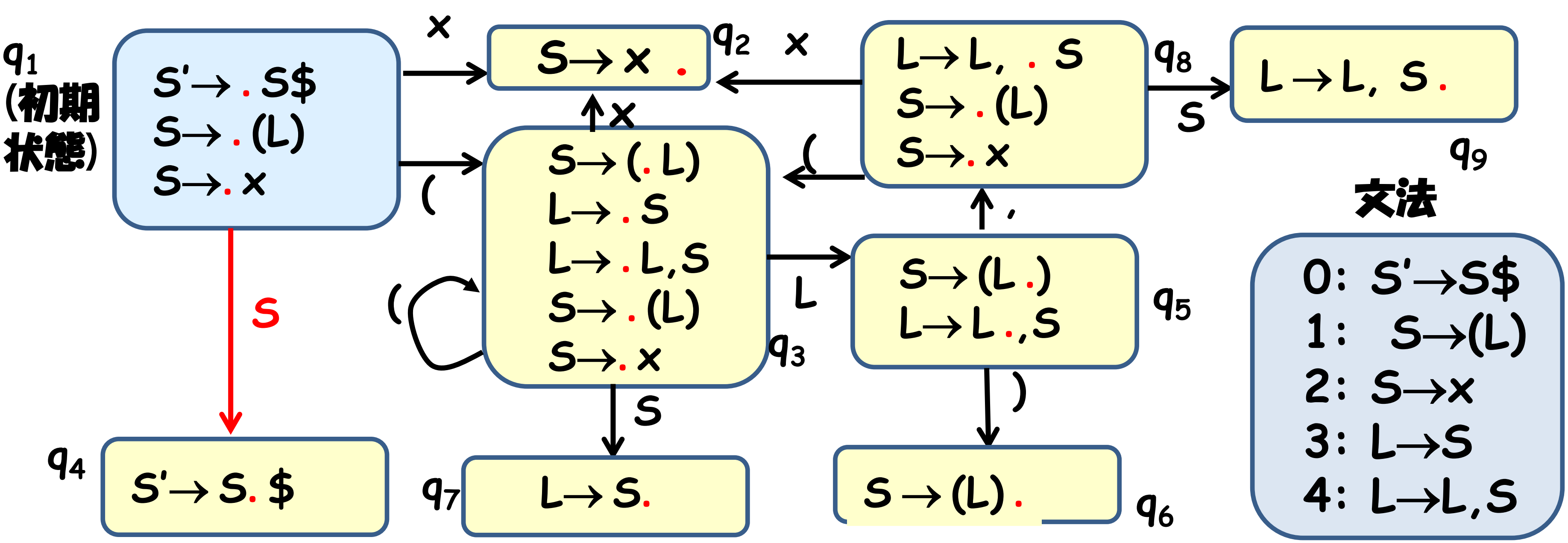
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3
$q_1(q_3 L q_5$	,x)$	shift
$q_1(q_3 L q_5 , q_8$	x)$	shift
$q_1(q_3 L q_5 , q_8 x q_2$	)$	reduce by 2
$q_1(q_3 L q_5 , q_8 S q_9$	)$	reduce by 4
$q_1(q_3 L q_5$	)$	shift
$q_1(q_3 L q_5) q_6$	$	reduce by 1



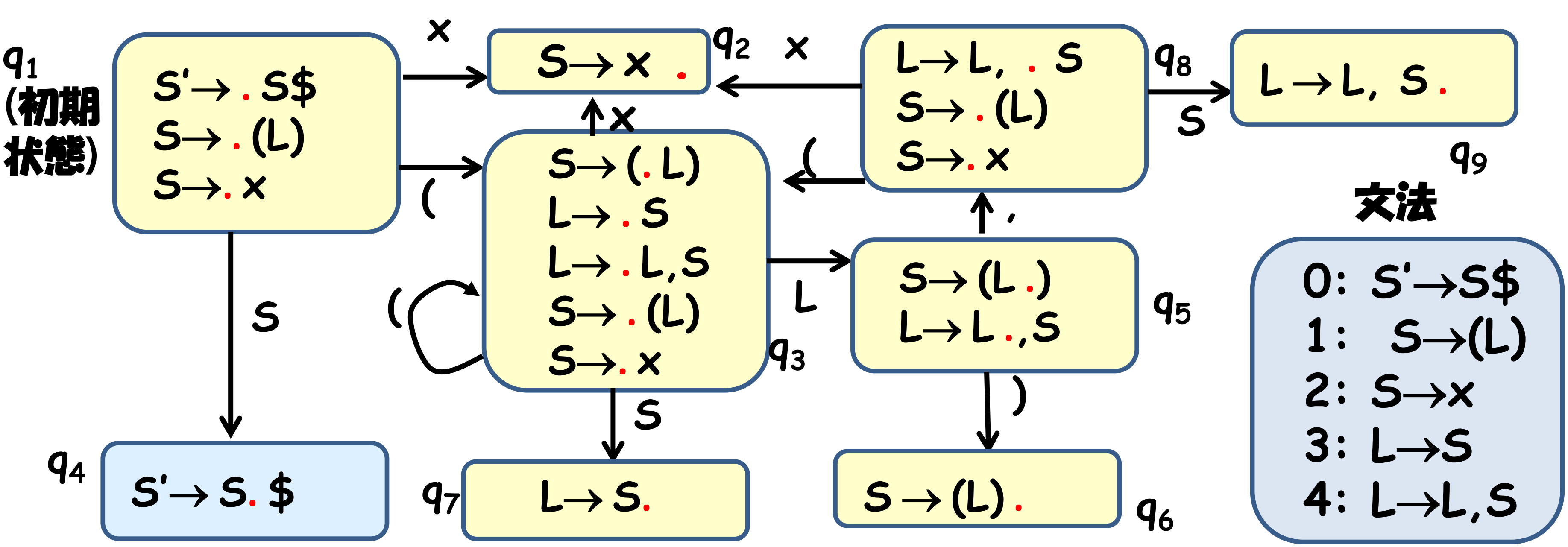
解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3xq_2$	,x)$	reduce by 2
$q_1(q_3Sq_7$	,x)$	reduce by 3
$q_1(q_3Lq_5$	,x)$	shift
$q_1(q_3Lq_5,q_8$	x)$	shift
$q_1(q_3Lq_5,q_8xq_2$	)$	reduce by 2
$q_1(q_3Lq_5,q_8Sq_9$	)$	reduce by 4
$q_1(q_3Lq_5$	)$	shift
$q_1(q_3Lq_5)q_6$	$	reduce by 1



解析例:

スタック	入力	アクション
q_1	(x,x)$	shift
$q_1(q_3$	x,x)$	shift
$q_1(q_3 x q_2$	,x)$	reduce by 2
$q_1(q_3 S q_7$	,x)$	reduce by 3
$q_1(q_3 L q_5$	,x)$	shift
$q_1(q_3 L q_5 , q_8$	x)$	shift
$q_1(q_3 L q_5 , q_8 x q_2$	)$	reduce by 2
$q_1(q_3 L q_5 , q_8 S q_9$	)$	reduce by 4
$q_1(q_3 L q_5$	)$	shift
$q_1(q_3 L q_5) q_6$	$	reduce by 1
$q_1 S q_4$	$	accept



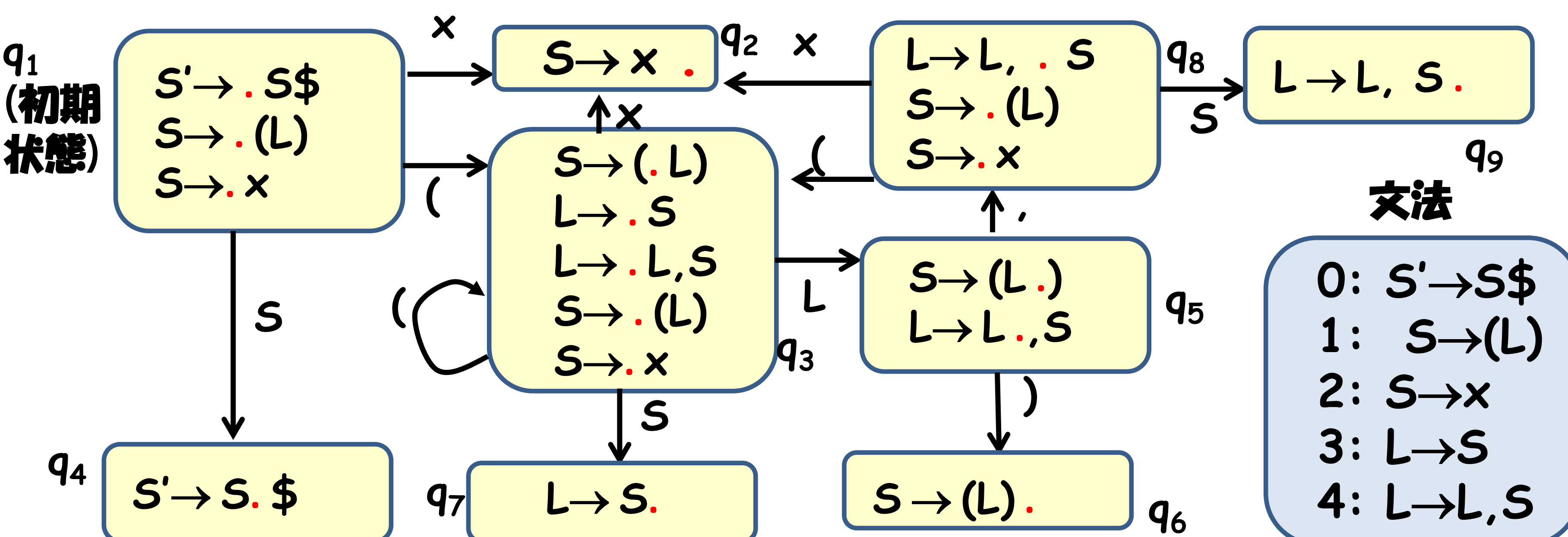
LR(0)構文解析表

- 以下の情報をあらかじめ計算して表に
  - 各LR(0)状態と入力文字に対するアクション (shift/reduce/accept/error)
  - 各LR(0)状態と非終端記号に対する遷移先
- 一つのエントリーに複数のアクションがあれば、LR(0)で解析不能(LR(0)文法ではない）

LR(0)構文解析表の例

LR(0)状態

	(	)	x	,	$	S	L
q_1	s3		s2			g4	
q_2	r2	r2	r2	r2	r2		
q_3	s3		s2			g7	g5
q_4					a		
q_5		s6		s8			
q_6	r1	r1	r1	r1	r1		
…			…				

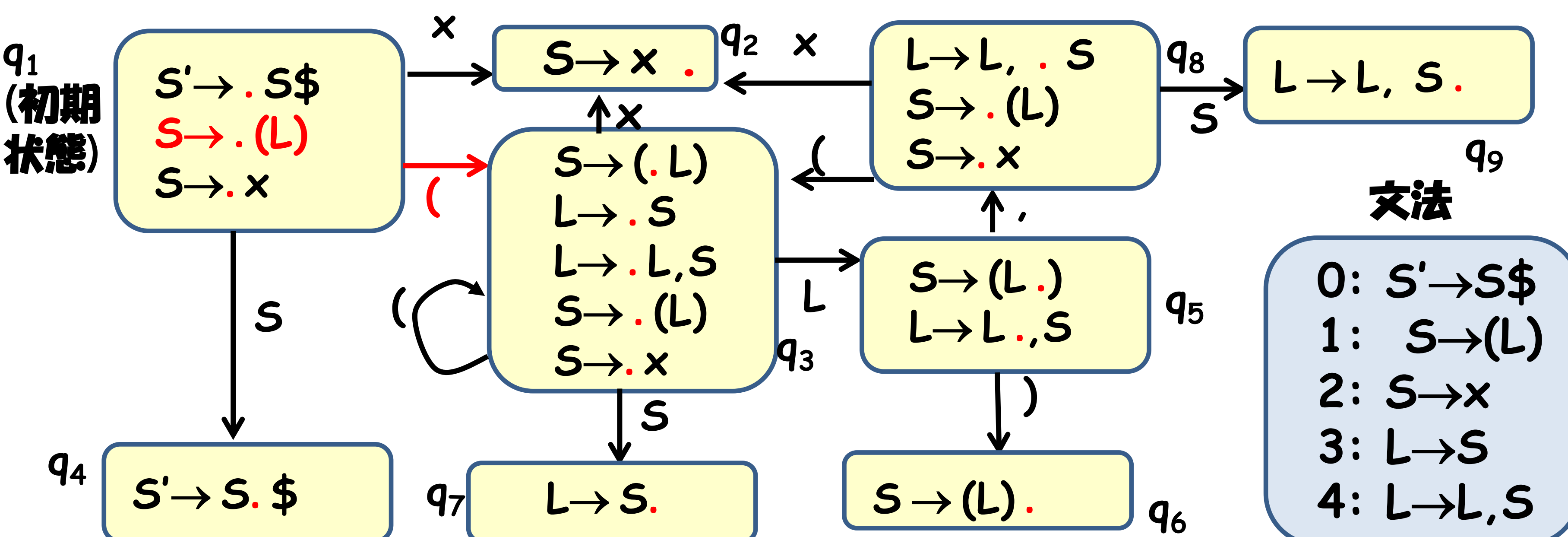


LR(0)構文解析表の例

shiftしてq_3に遷移せよ

LR(0)状態

	(	)	x	,	$	S	L
q_1	s3		s2			g4	
q_2	r2	r2	r2	r2	r2		
q_3	s3		s2			g7	g5
q_4					a		
q_5		s6		s8			
q_6	r1	r1	r1	r1	r1		
…			…				



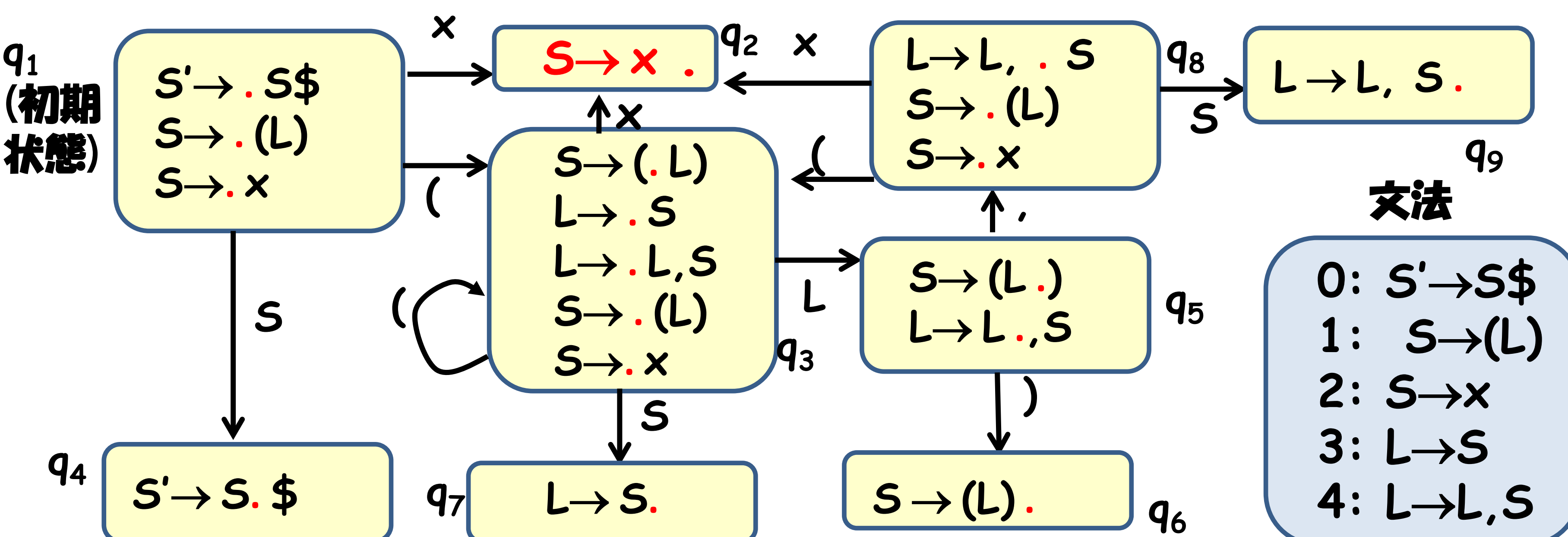
LR(0)構文解析表の例

shiftしてq_3に遷移せよ

規則2でreduce せよ

LR(0)状態

	(	)	x	,	$	S	L
q_1	s3		s2			g4	
q_2	r2	r2	r2	r2	r2		
q_3	s3		s2			g7	g5
q_4					a		
q_5		s6		s8			
q_6	r1	r1	r1	r1	r1		
…			…				



LR(0)構文解析表の例

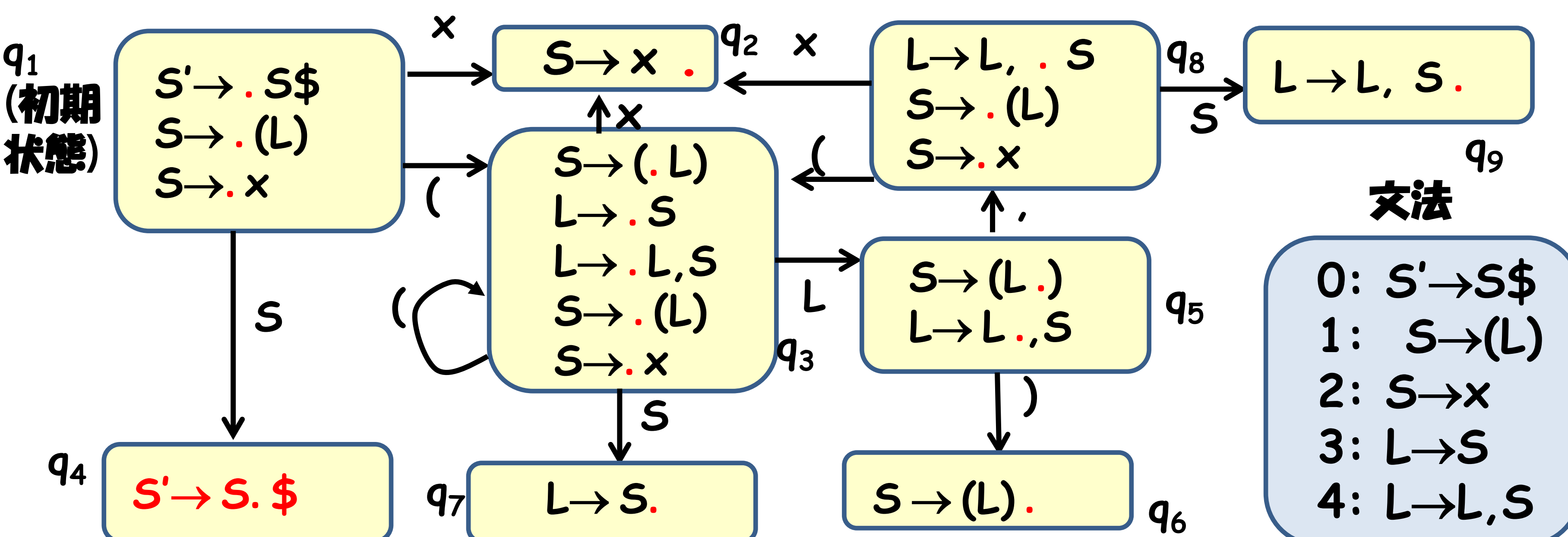
shiftしてq3に遷移せよ

規則2でreduce せよ

LR(0)状態

	(	)	x	,	$	S	L
q_1	s3		s2			g4	
q_2	r2	r2	r2	r2	r2		
q_3	s3		s2			g7	g5
q_4					a		
q_5		s6		s8			
q_6	r1	r1	r1	r1	r1		
…			…				

accept



LR(0)構文解析表の例

shiftしてq3に遷移せよ

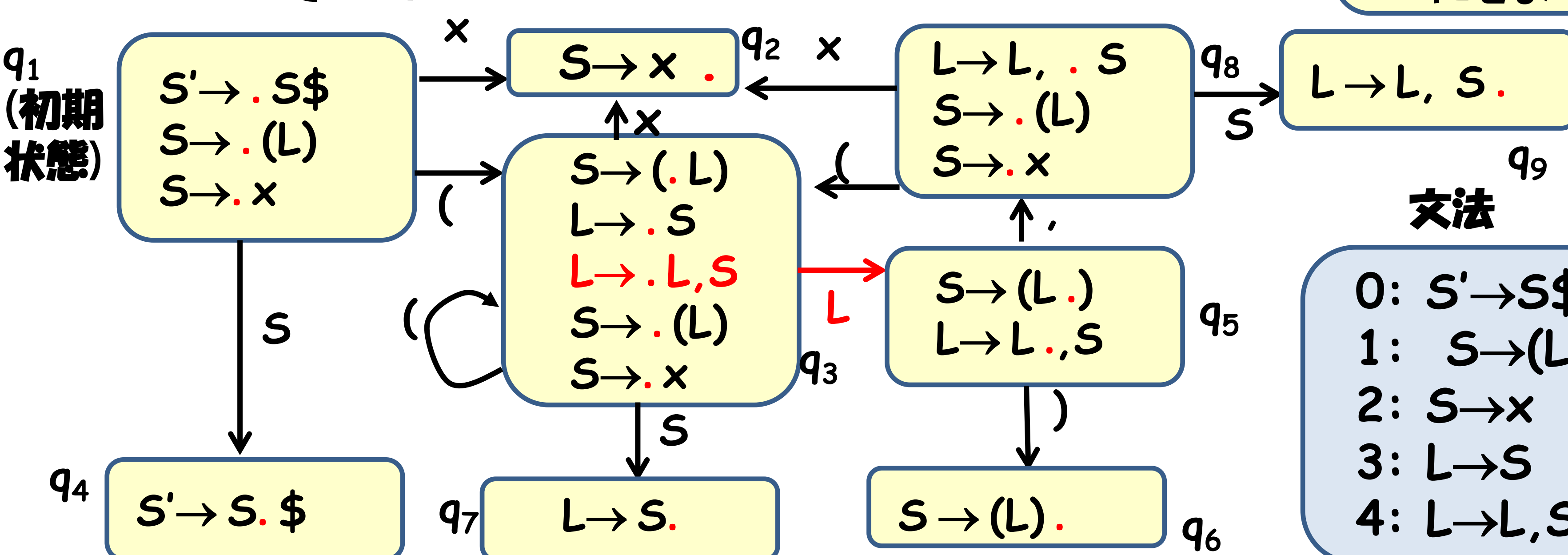
規則2でreduce せよ

LR(0)状態

	(	)	x	,	$	S	L
q_1	s3		s2			g4	
q_2	r2	r2	r2	r2	r2		
q_3	s3		s2			g7	g5
q_4					a		
q_5		s6		s8			
q_6	r1	r1	r1	r1	r1		
…			…				

accept

LをスタックにプッシュしたらLR(0)状態をq_5にせよ



文法

0: S'→S$
1: S→(L)
2: S→x
3: L→S
4: L→L,S

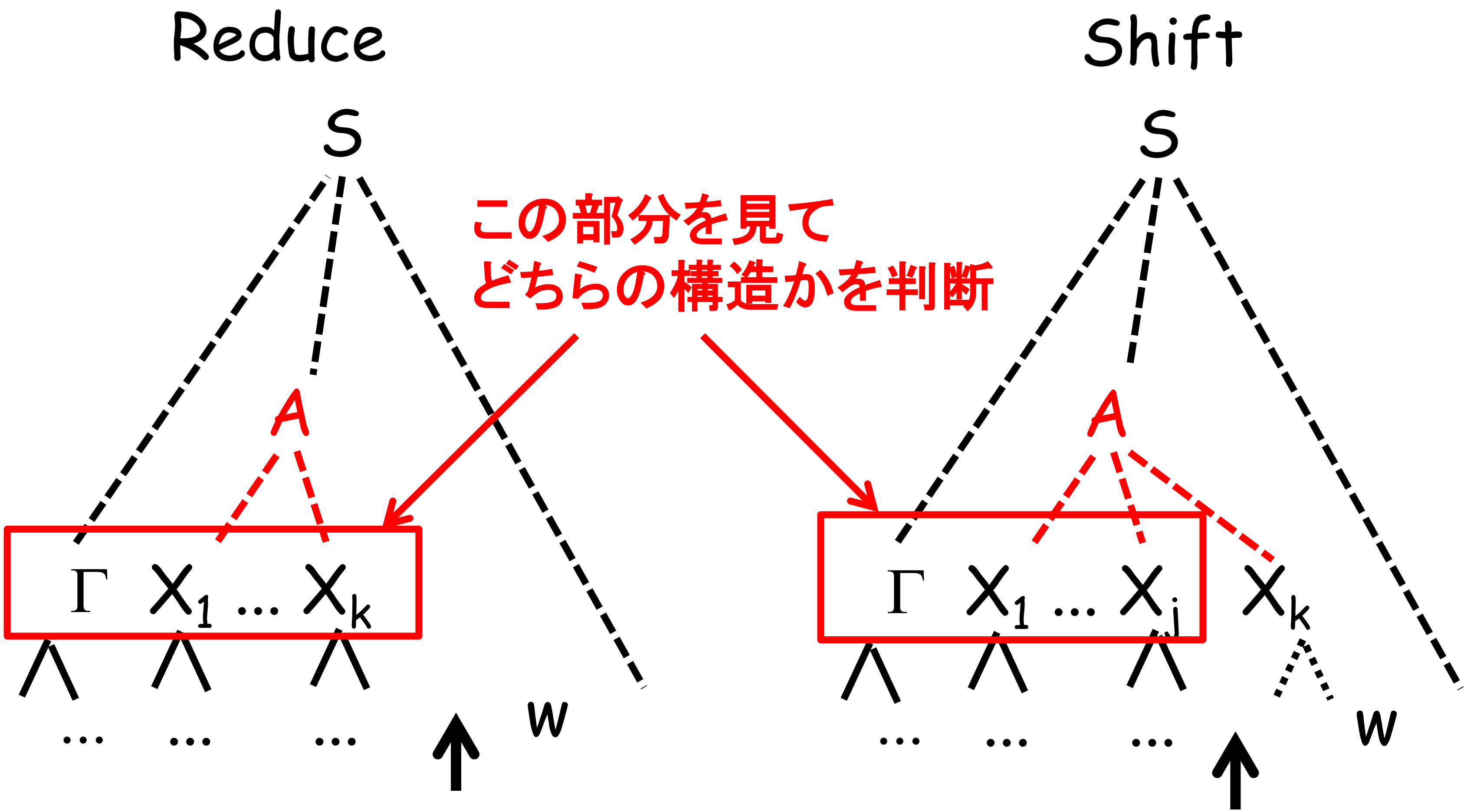
LR(0)解析アルゴリズム

```
main w = parse(q0,  w$)
parse (Γq, aw) =
   match  lookup_LRtable(q, a) with
      Shift(q') ->  parse (Γqaq', w)
     | Reduce(A->X1...Xn) ->
          let Γ'q' = pop(Γ, n) in
          let Goto(q'') = lookup_LRtable(q', A) in
             parse(Γ'q'Aq'', aw)
     | Accept -> finish()
     | Empty -> report_error()
```

アウトライン

- LR構文解析
- LR(0)
  - shift/reduceの判断法
  - LR(0)オートマトン
  - LR(0)アルゴリズムの実際
  - 形式言語理論の立場からみたLR(0)の仕組み
- LR(0)の改良版
  - SLR
  - LR(k)
  - LALR

Shift vs Reduce



最右導出とshift/reduce

$S' \to S\ .\ \$$

$\to (L)\ .\ \$$

$\to (L\ .\ )\$$

$\to (L,S\ .\ )\$$

$\to (L,x\ .\ )\$$

$\to (L,\ .\ x\ )\$$

$\to (L\ .,\ x\ )\$$

$\to (S\ .,\ x\ )\$$

$\to (x\ .,\ x\ )\$$

$\to (.x,\ x\ )\$$

$\to .\ (x,\ x\ )\$$

スタックが$\Gamma\alpha_1\ldots\alpha_n$のときに
$A \to \alpha_1\ldots\alpha_n$
でreduceして解析が成功する可能性がある
<-->
$S \to^* \Gamma A w \to \Gamma\alpha_1\ldots\alpha_n w$
の形の最右導出が存在

スタックが$\Gamma\alpha_1\ldots\alpha_n$、残りの入力の先頭が
cのときshiftして解析が成功する可能性がある
<-->
規則$A \to \alpha_1\ldots\alpha_n c\beta_1\ldots\beta_m$が存在し、
$S \to^* \Gamma A w \to \Gamma\alpha_1\ldots\alpha_n c\beta_1\ldots\beta_m w$
の形の最右導出が存在

$S' \to S\$ \quad S \to (L) \quad S \to x \quad L \to S \quad L \to L,S$

最右導出とshift/reduce

Left(A) = $\{\Gamma \mid S \to^* \Gamma A w\}$
とすると…

スタックが$\Gamma\alpha_1\ldots\alpha_n$のときに
$A \to \alpha_1\ldots\alpha_n$
でreduceして解析が成功する可能性がある
<-->
$S \to^* \Gamma A w \to \Gamma\alpha_1\ldots\alpha_n w$
の形の最右導出が存在

$\to$ (L, . x)$
$\to$ (L ., x)$
$\to$ (S ., x)$
$\to$ (x ., x)$
$\to$ (.x, x)$
$\to$. (x, x)$

スタックが$\Gamma\alpha_1\ldots\alpha_n$、残りの入力の先頭が
cのときshiftして解析が成功する可能性がある
<-->
規則$A \to \alpha_1\ldots\alpha_n c\beta_1\ldots\beta_m$が存在し、
$S \to^* \Gamma A w \to \Gamma\alpha_1\ldots\alpha_n c\beta_1\ldots\beta_m w$
の形の最右導出が存在

$S' \to S\$$　$S \to (L)$　$S \to x$　$L \to S$　$L \to L,S$

最右導出とshift/reduce

Left(A) =
{Γ |
$S \to^* \Gamma A w$ }
とすると...

スタックが$\Gamma\alpha_1\ldots\alpha_n$のときに
$A \to \alpha_1\ldots\alpha_n$
でreduceして解析が成功する可能性がある
<-->
$\Gamma \in$ Left(A)

$\to$ (L, . x)$
$\to$ (L ., x)$
$\to$ (S ., x)$
$\to$ (x ., x)$
$\to$ (.x, x)$
$\to$. (x, x)$

スタックが$\Gamma\alpha_1\ldots\alpha_n$、残りの入力の先頭が
cのときshiftして解析が成功する可能性がある
<-->
規則$A \to \alpha_1\ldots\alpha_n c\beta_1\ldots\beta_m$が存在し、
$S \to^* \Gamma A w \to \Gamma\alpha_1\ldots\alpha_n c\beta_1\ldots\beta_m w$
の形の最右導出が存在

$S' \to S\$$ $S \to (L)$ $S \to x$ $L \to S$ $L \to L,S$

最右導出とshift/reduce

Left(A) =
{Γ |
 S $\rightarrow^*\Gamma$Aw }
とすると…

スタックが$\Gamma\alpha_1 \ldots \alpha_n$のときに
A $\rightarrow \alpha_1 \ldots \alpha_n$
でreduceして解析が成功する可能性がある
<-->
 $\Gamma \in$ Left(A)

$\rightarrow$ (L, . x)$
$\rightarrow$ (L ., x)$
$\rightarrow$ (S ., x)$
$\rightarrow$ (x ., x)$
$\rightarrow$ (.x, x)$
$\rightarrow$. (x, x)$

スタックが$\Gamma\alpha_1 \ldots \alpha_n$、残りの入力の先頭が
cのときshiftして解析が成功する可能性がある
<-->
規則A $\rightarrow \alpha_1 \ldots \alpha_n c \beta_1 \ldots \beta_m$が存在し、
 $\Gamma \in$ Left(A)

S'$\rightarrow$S$　S$\rightarrow$(L)　S$\rightarrow$x　L$\rightarrow$S　L$\rightarrow$L,S

最右導出とshift/reduce

Left(A) =
{Γ |
$S \to^* \Gamma A w$ }
とすると…

スタックが$\Gamma\alpha_1\ldots\alpha_n$のときに
$A \to \alpha_1\ldots\alpha_n$
でreduceして解析が成功する可能性がある
<-->
$\Gamma \in$ Left(A)

スタックが$\Gamma\alpha_1\ldots\alpha_n$、残りの入力の先頭が
cのときshiftして解析が成功する可能性がある
<-->
規則$A \to \alpha_1\ldots\alpha_n c \beta_1\ldots\beta_m$が存在し、
$\Gamma \in$ Left(A)

shift/resetの判定手続き:
if $\Gamma \in$ Left(A) $\alpha_1\ldots\alpha_n$ for some $A \to \alpha_1\ldots\alpha_n$ then
  reduce by $A \to \alpha_1\ldots\alpha_n$
else if $\Gamma \in$ Left(A) $\alpha_1\ldots\alpha_n$ for some $A \to \alpha_1\ldots\alpha_n cw$ and next=c then
  shift
else error

最右導出とshift/reduce

Left(A) = {Γ | S $\rightarrow^* \Gamma A w$ } とすると...

shift/resetの判定手続き:

if $\Gamma \in$ Left(A) $\alpha_1 \ldots \alpha_n$ for some $A \rightarrow \alpha_1 \ldots \alpha_n$ then
 reduce by $A \rightarrow \alpha_1 \ldots \alpha_n$
else if $\Gamma \in$ Left(A) $\alpha_1 \ldots \alpha_n$ for some $A \rightarrow \alpha_1 \ldots \alpha_n c w$ and next=c then
 shift
else error

Left(A)は正規言語なので決定可能！

LR(0)オートマトンはLeft(A) $\alpha_1 \ldots \alpha_n$を読むと
[$A \rightarrow \alpha_1 \ldots \alpha_n . \beta_1 \ldots \beta_m$]を含む状態に遷移

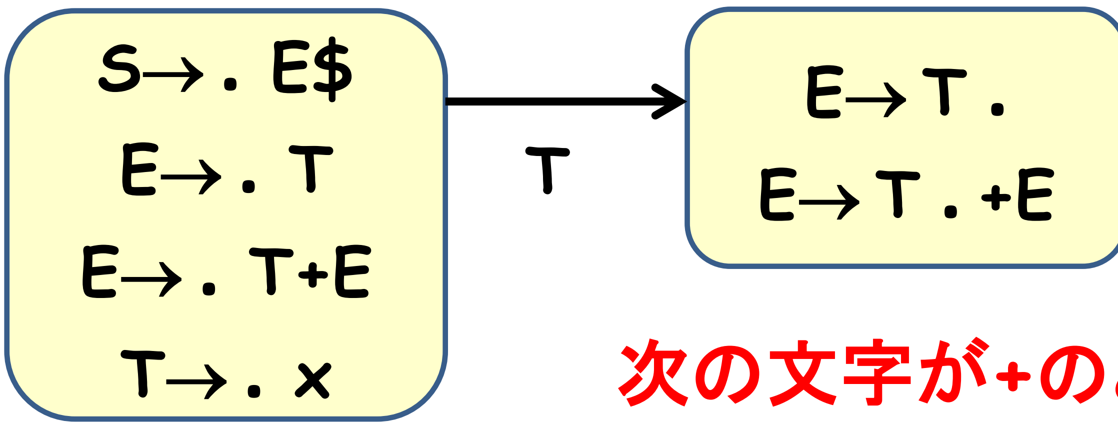
アウトライン

- LR構文解析
- LR(0)
  - shift/reduceの判断法
  - LR(0)オートマトン
  - LR(0)アルゴリズムの実際
  - 形式言語理論の立場からみたLR(0)の仕組み
- LR(0)の改良版
  - SLR
  - LR(k)
  - LALR

LR(0)の欠点

- 扱える文法クラスが狭い
  （先読みをしないため）

例：{S→E$,　E→T,　E→T+E,　T→x}

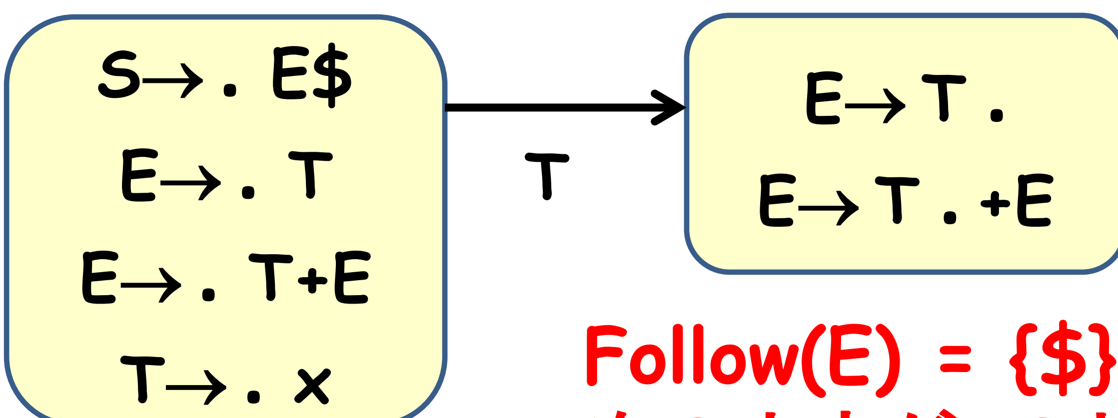


次の文字が+のときに
reduceとshift がconflict

SLR(1) （“(1)“はしばしば省略）

- **LR(0)オートマトンを用いるが、LLで用いた“Follow“集合を用いてreduceを制限**
  - **reduce を行う条件：**
    - **$A \to \alpha$ ． がLR(0)状態に含まれ、かつ**
    - **先読みシンボルc が Follow(A)に含まれる**

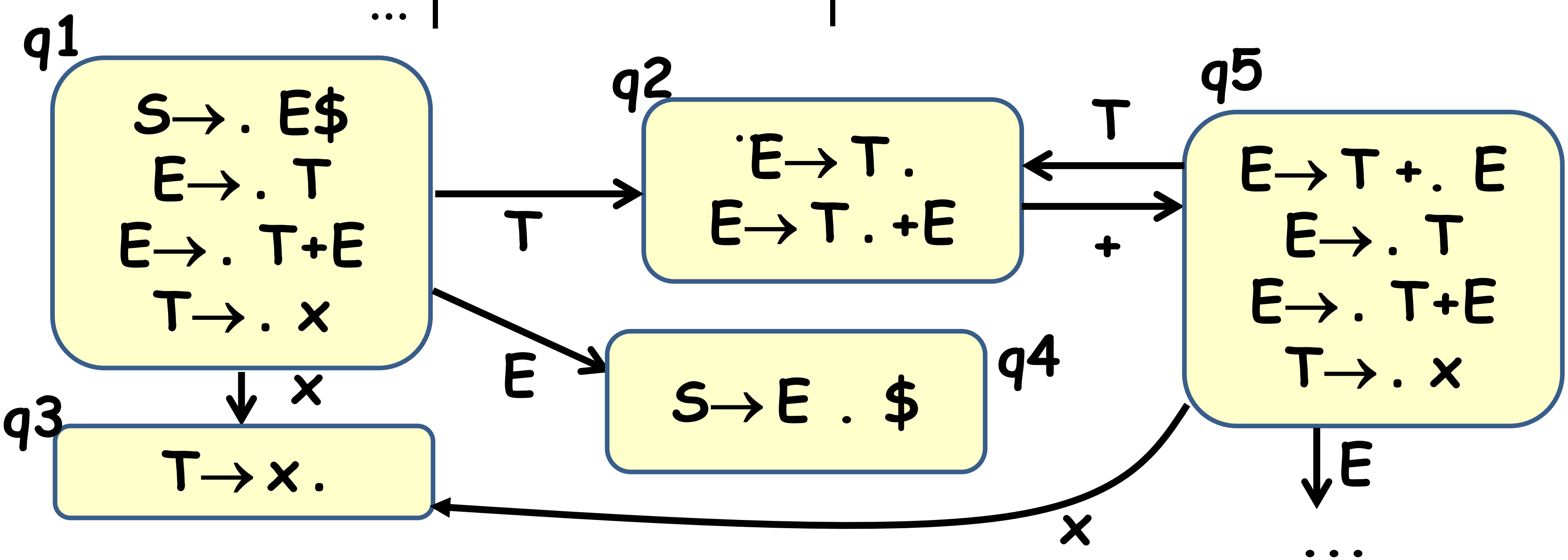
例：{S→E$,　E→T,　E→T+E,　T→x}



Follow(E) = {$}なので、
次の文字が+のときはshiftのみ！

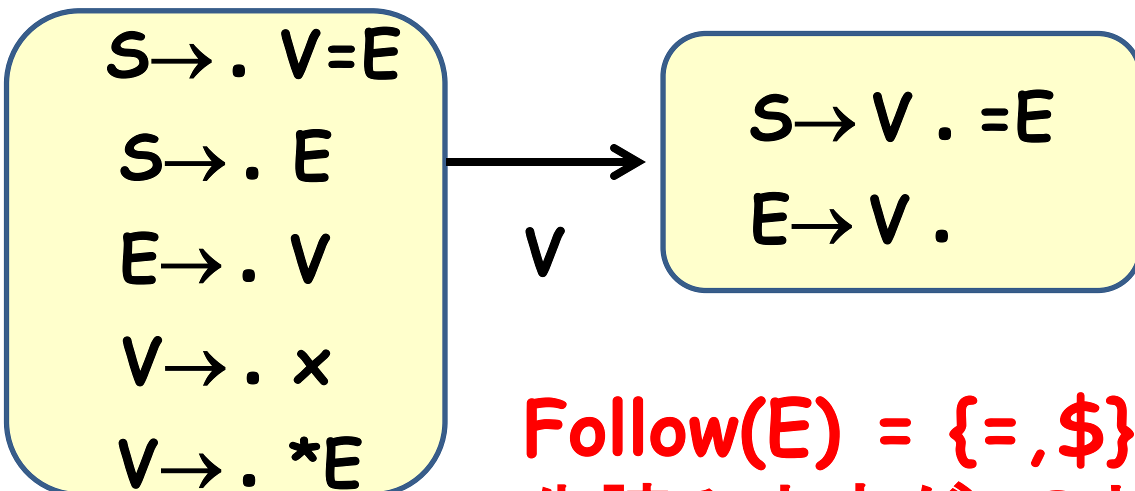
{0:S→E$ 1:E→T 2:E→T+E 3:T→x}の SLR(1)構文解析表

	x	+	$	S	E	T
q_1	s3				g4	g2
q_2		s5	r1			
q_3		r3	r3			
q_4			a			
…						



SLR(1)で解析できない文法

{S'→S$, S→V=E, S→E,
E→V, V→x, V→*E}



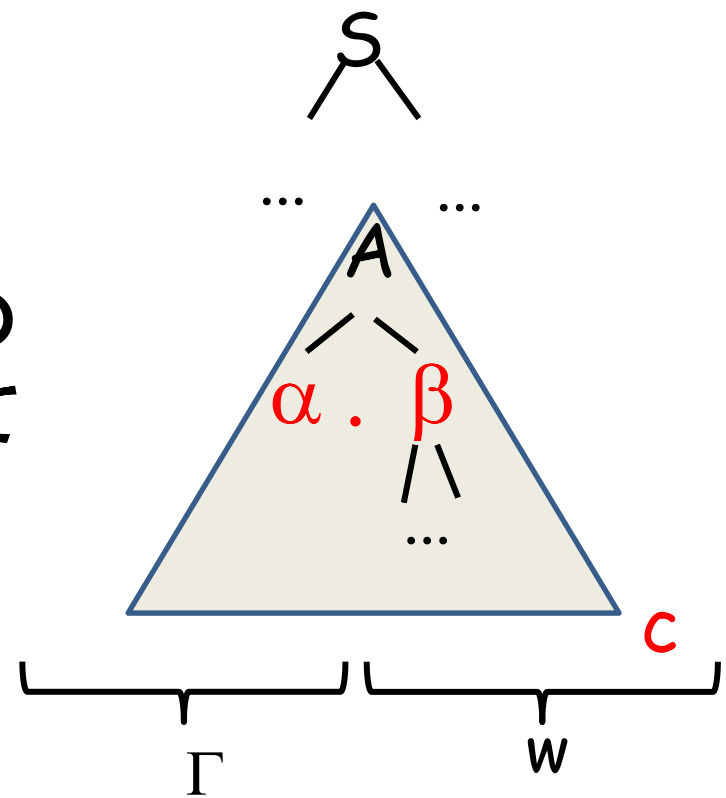
Follow(E) = {=,$}なので、
先読み文字が=のときに
shift/reduce conflict!

LR(1)

- LR(0)アイテムを拡張してreduce の判断を精緻化
- LR(1)アイテム:

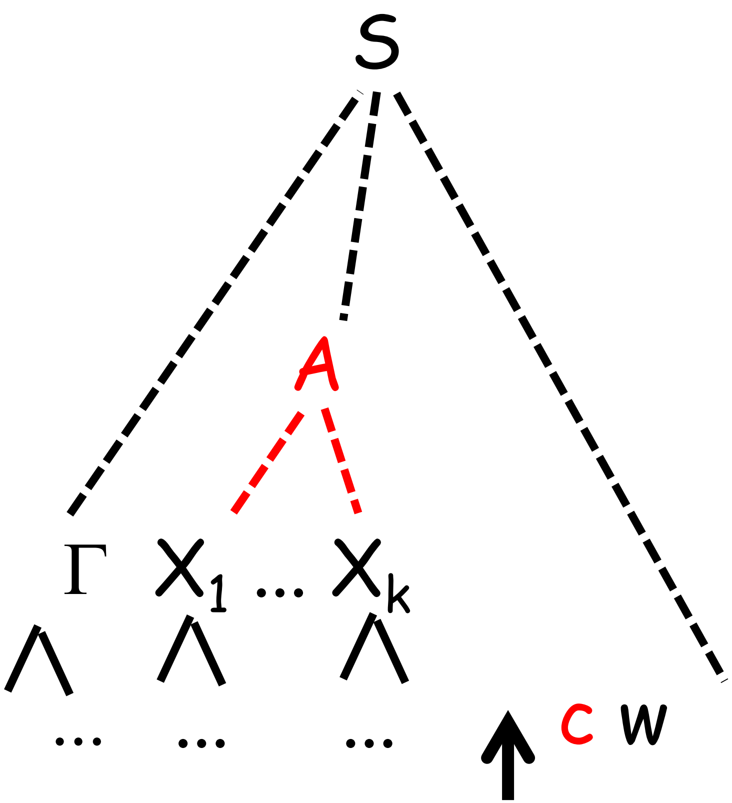
  $[A \rightarrow \alpha .\ \beta,\ c]$

  c： Aから生成される語の後ろに来ると期待されている文字



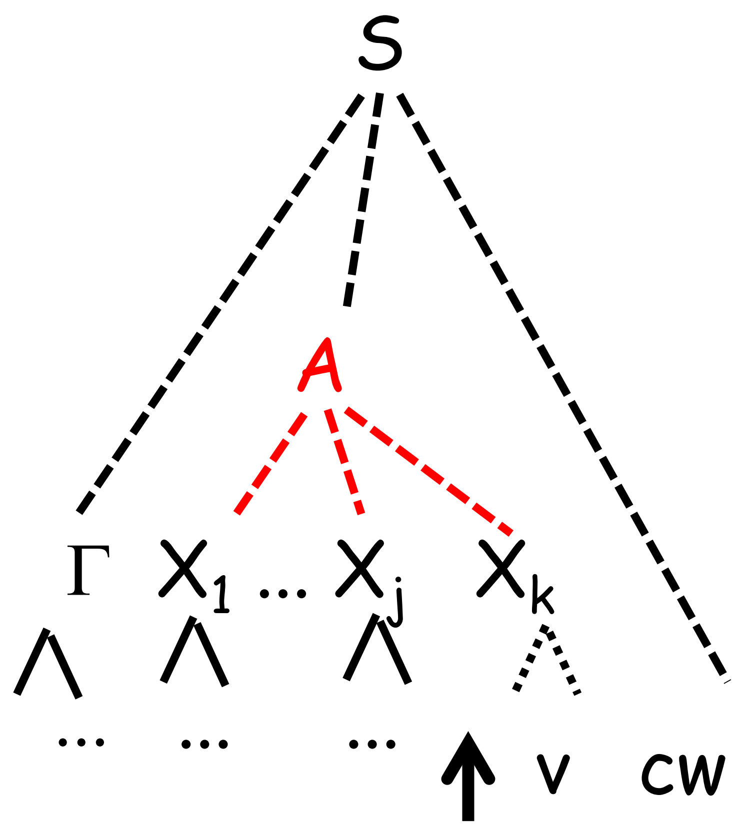
LR(1)に基づくshift/Reduceの選択

Reduce



$[A \to X_1 \ldots X_k\ .\ ,\ c\ ]$

Shift



$[A \to X_1 \ldots X_j\ .\ X_{j+1} \ldots X_k,\ c]$

LR(1)オートマトン

- LR(1)状態： LR(1)アイテムの集合qでclosure(q)=qを満たすもの
  - closure(q) =
    q⊆ q'かつ
    「[A → α. B β, c] ∈ q',
    B → γ ∈ G, d ∈FIRST(βc)
    ならば[B → . γ, d] ∈ q' 」を満たす最小の集合q'
- 初期状態： q_0 = closure({[S → . α, ?]})
- 遷移関数：

  δ(q, X) =
  closure({ [A → α X . β , c] | [A → α . X β, c] ∈ q})

例：{S'→S$, S→V=E, S→E, E→V, V→x, V→*E}

- S' → . S$?
- S → . V=E $
- S → . E $
- E → . V $
- V → . x =$
- V → . *E =$

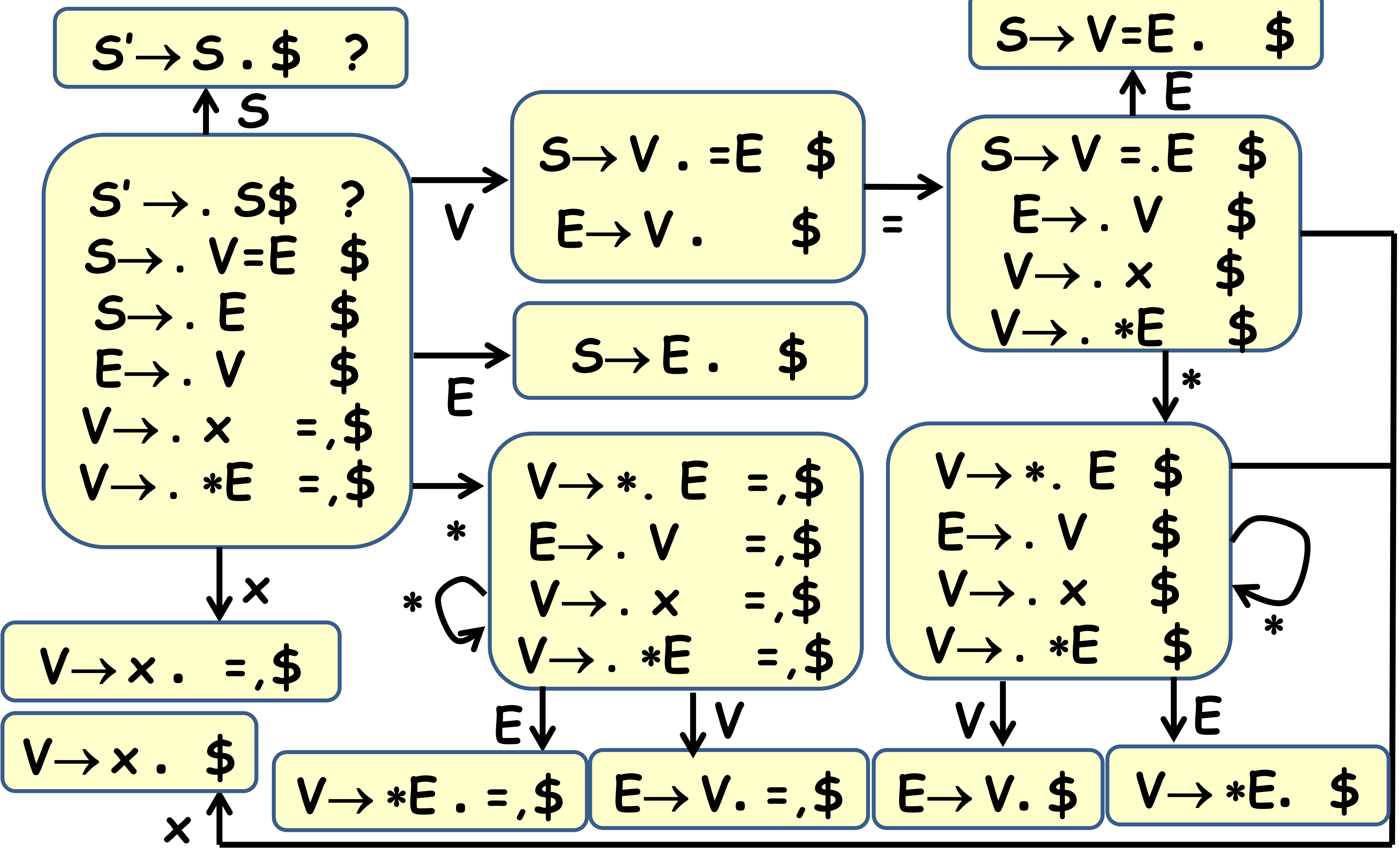
V

- S → V . =E $
- E → V . $

先読み文字が =: shift

$: reduce

例：{S'→S$, S→V=E, S→E, E→V, V→x, V→*E}



LALR(1)

- LR(1)状態のうち、先読み文字についてのみ異なるものをマージ（状態数を減らすため）

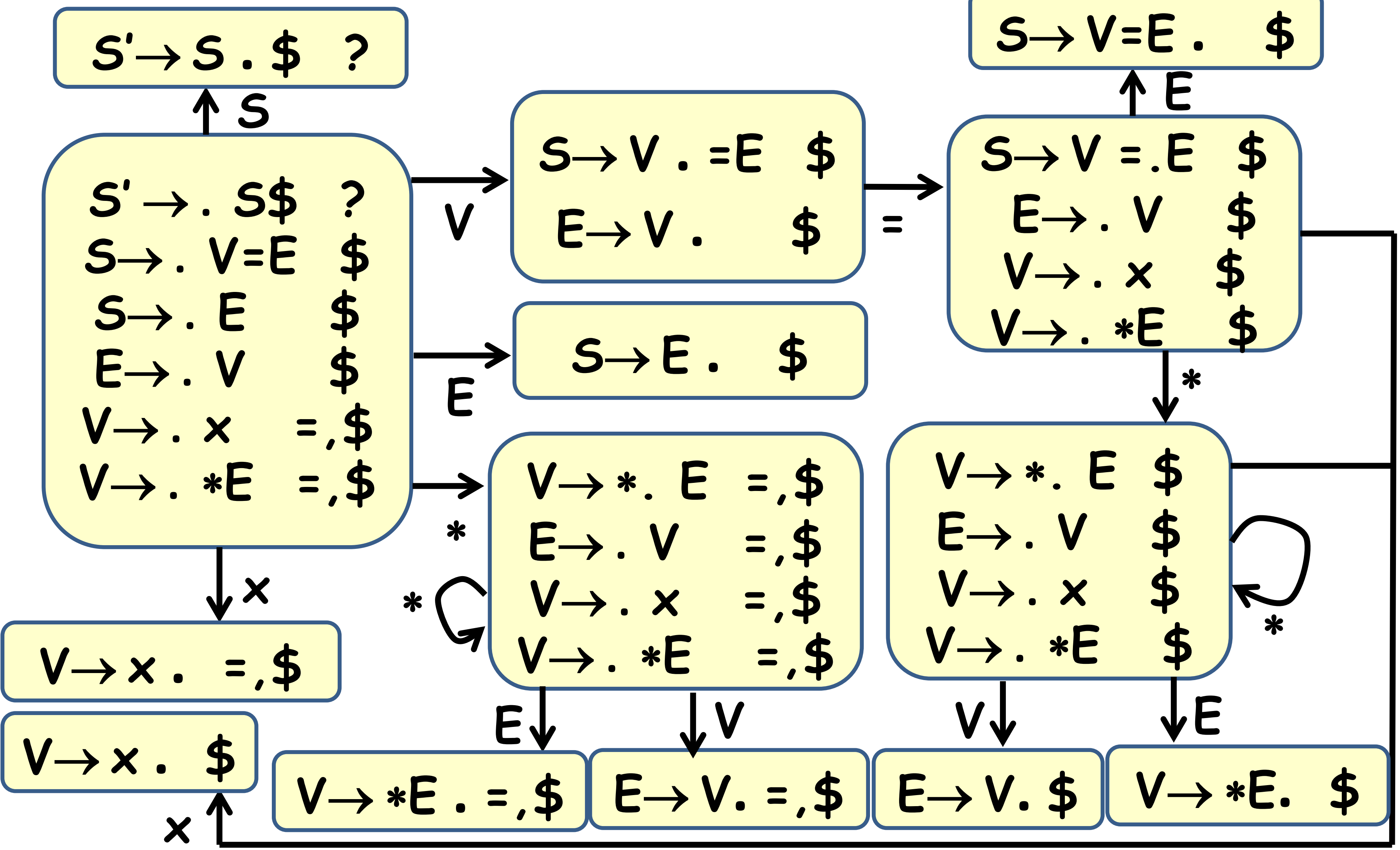
$V \to x.\quad =$

$V \to x.\quad \$$

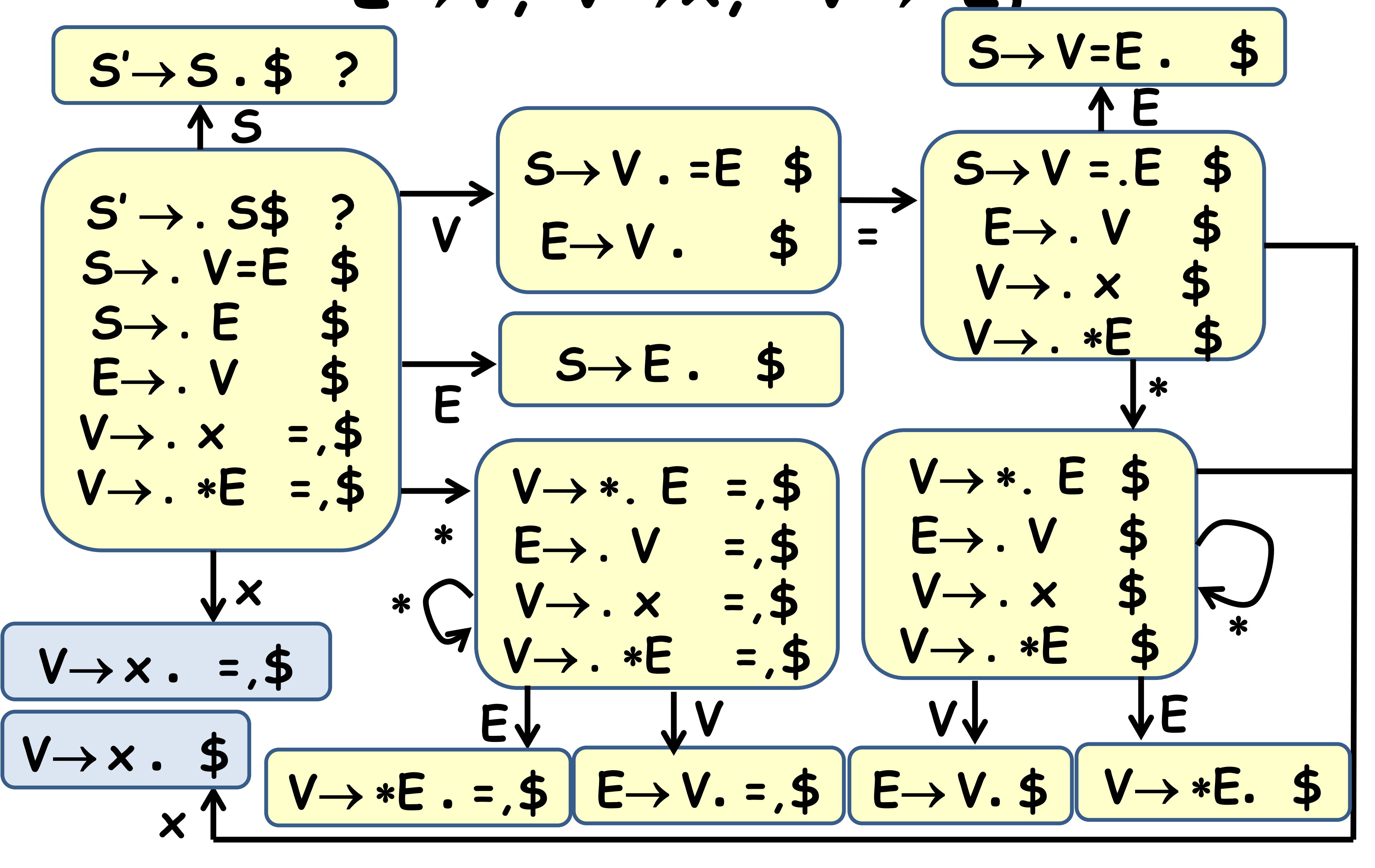
$\Rightarrow$ $V \to x.\quad =,\$$

- ほとんどのプログラミング言語の文法を扱うのに十分な表現力
- 構文解析器自動生成器yacc で採用

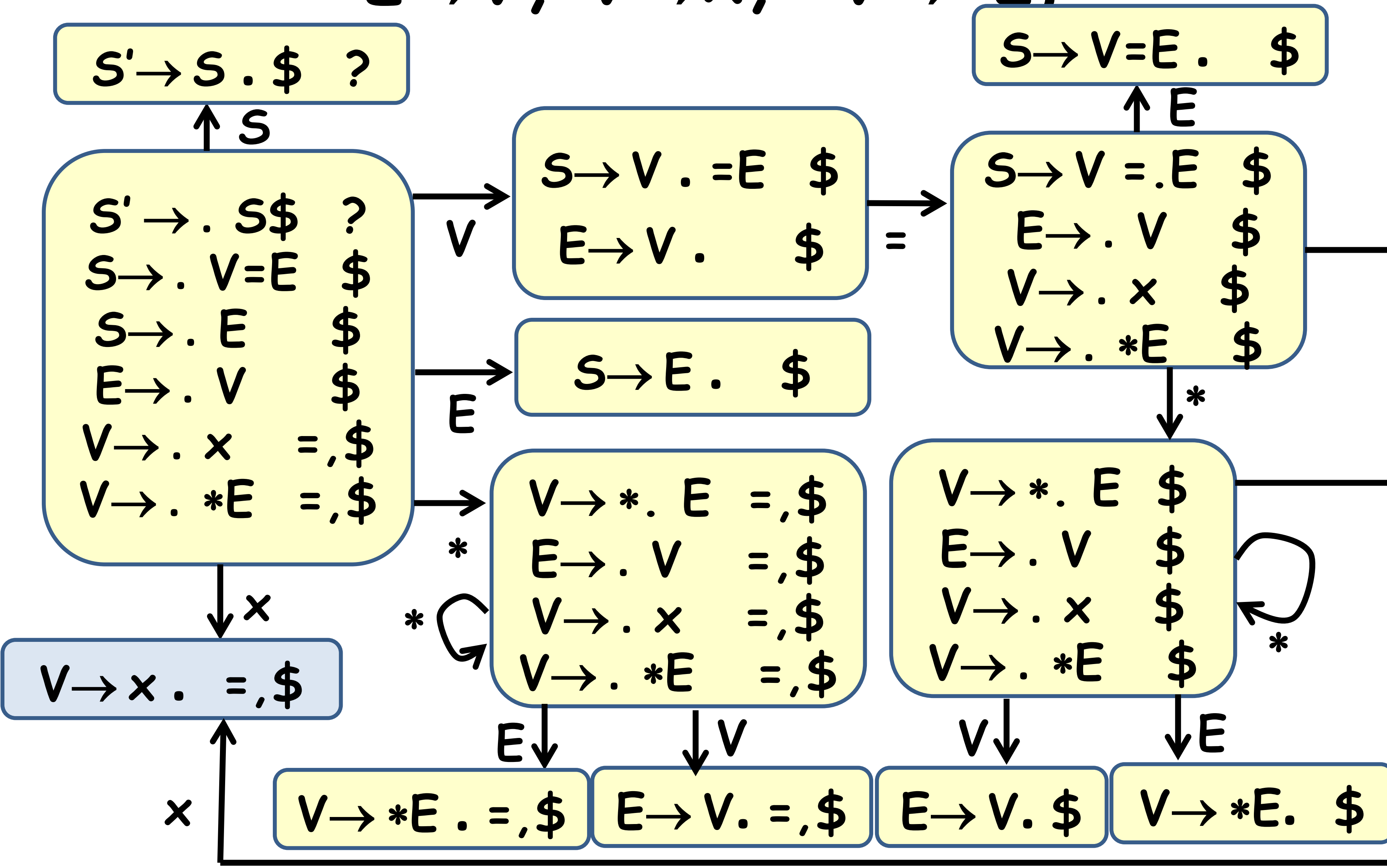
例：{S'→S$, S→V=E, S→E, E→V, V→x, V→*E}



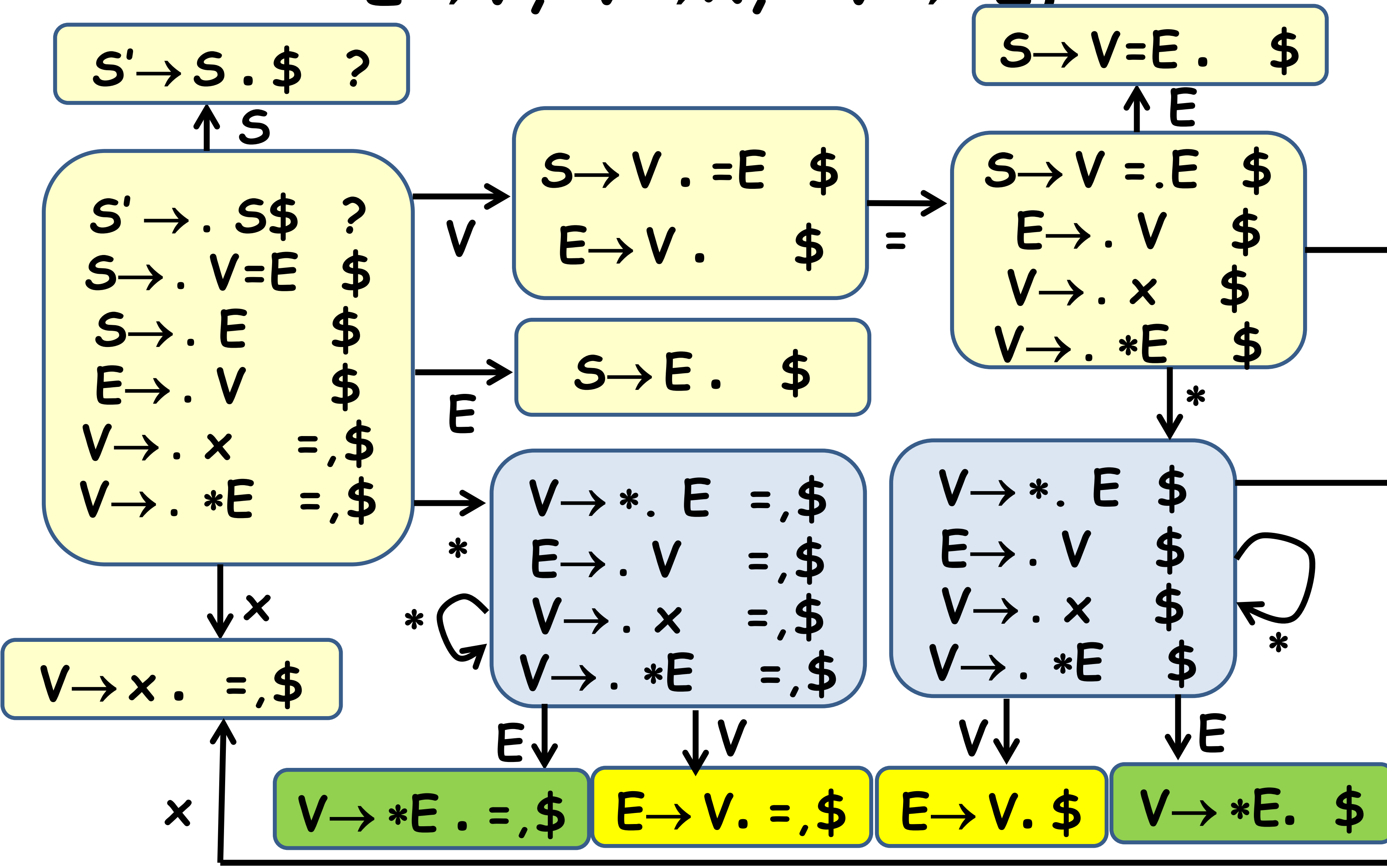
例：{S'→S$, S→V=E, S→E, E→V, V→x, V→*E}



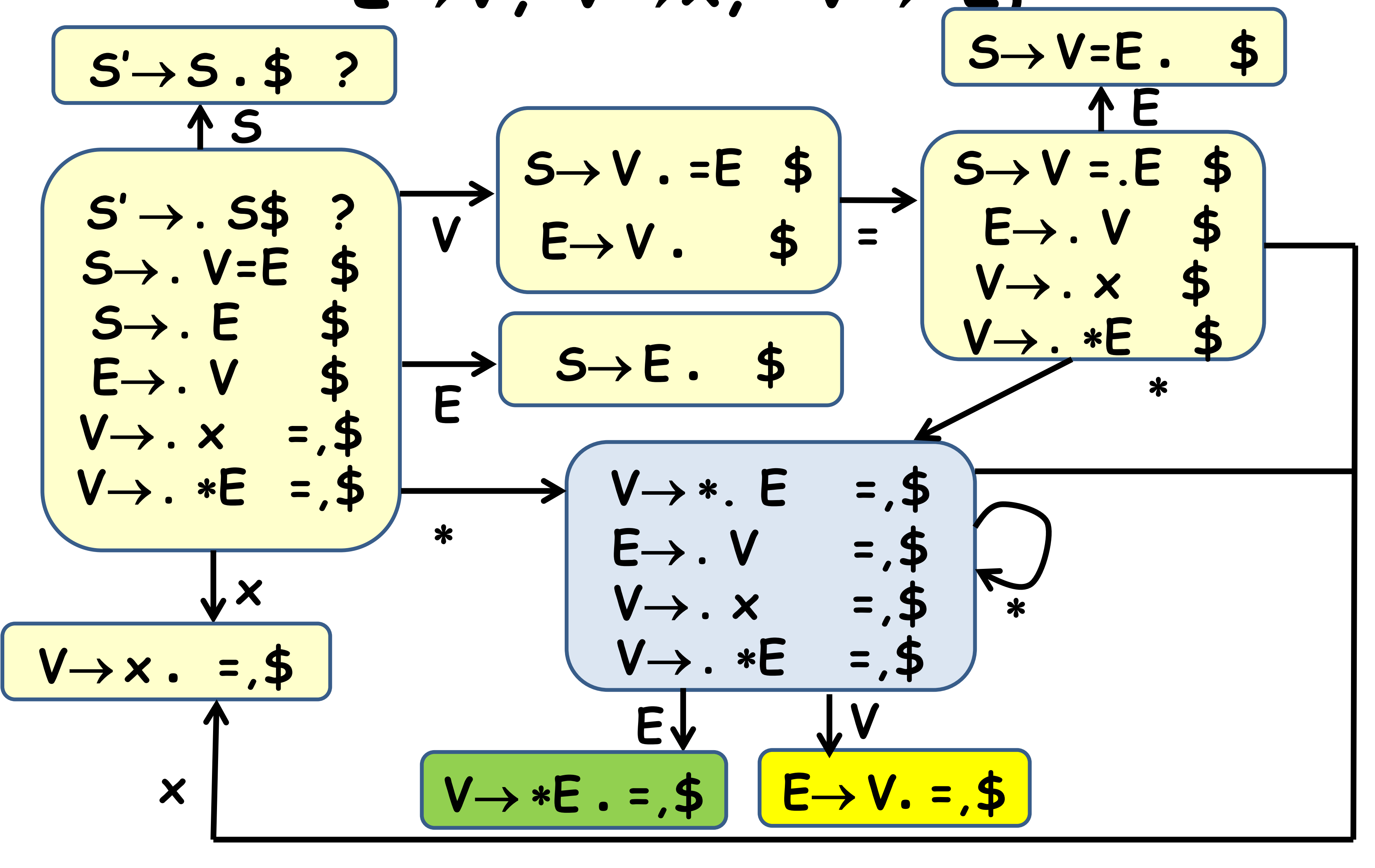
例：{S'→S$, S→V=E, S→E, E→V, V→x, V→*E}



例：{S'→S$, S→V=E, S→E, E→V, V→x, V→*E}

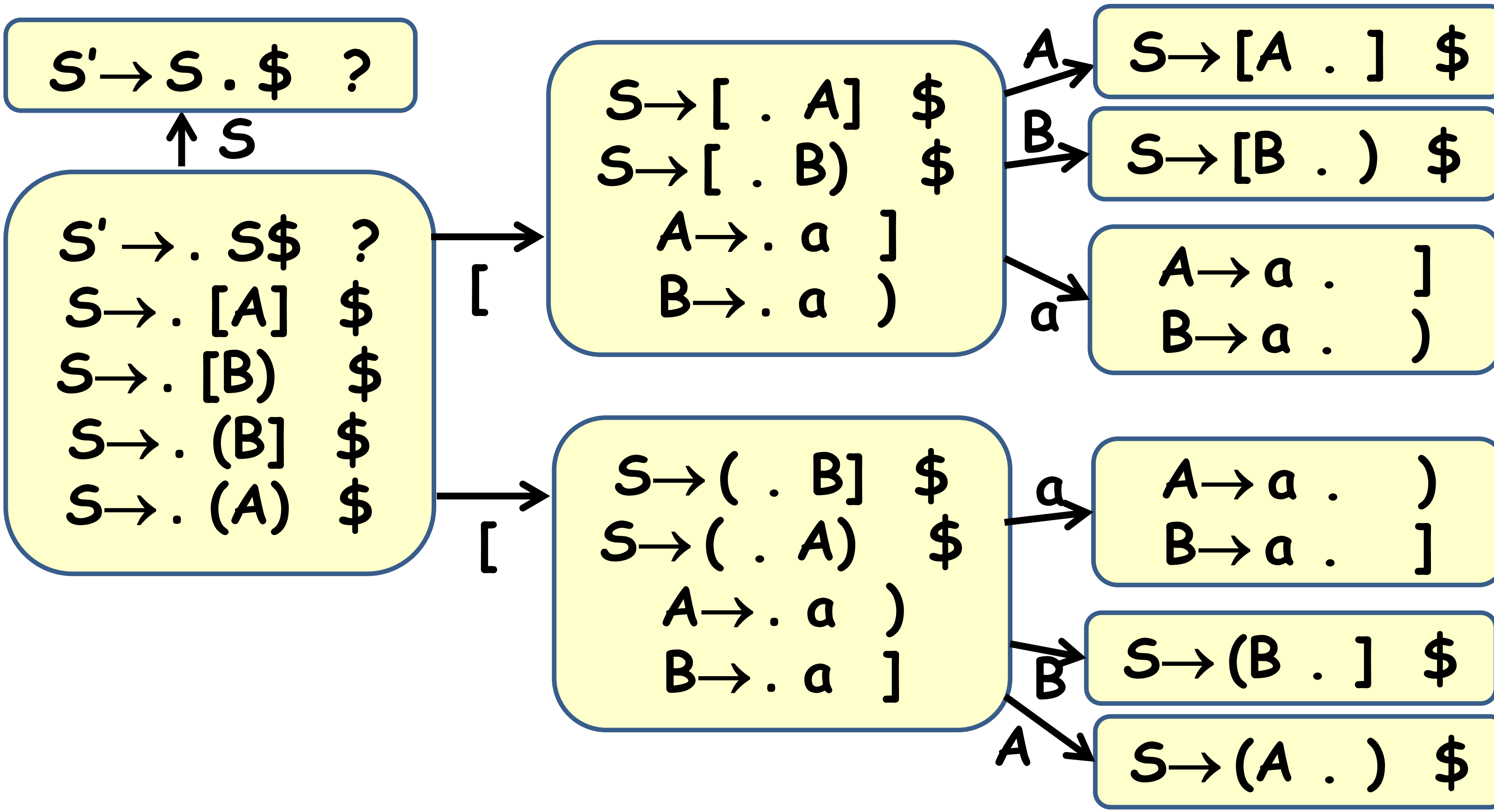


例：{S'→S$, S→V=E, S→E, E→V, V→x, V→*E}



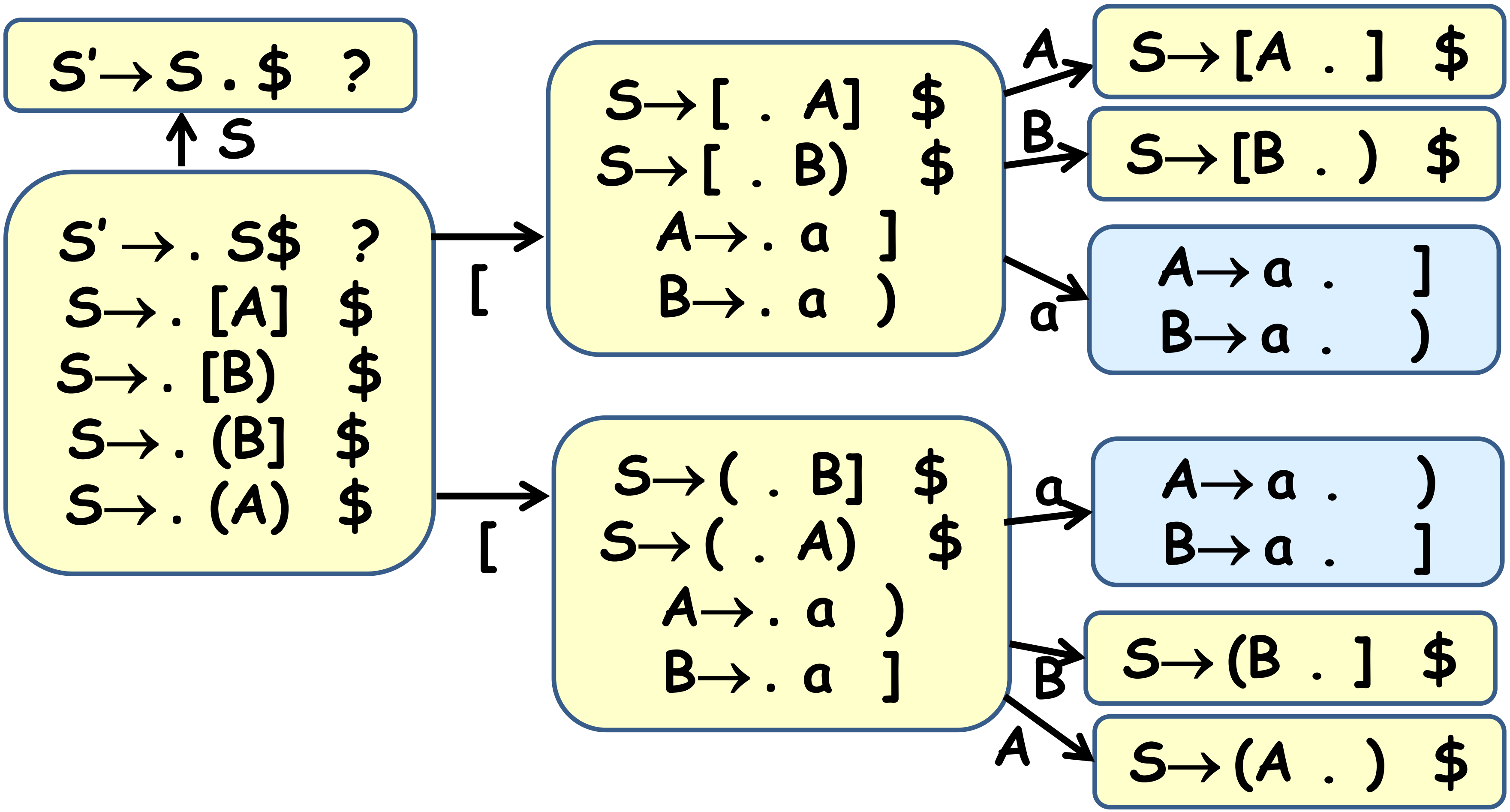
LR(1)で扱えてLALR(1)で扱えない文法

{S'→S$, S→[A] | [B) | (B] | (A),
A →a, B →a}



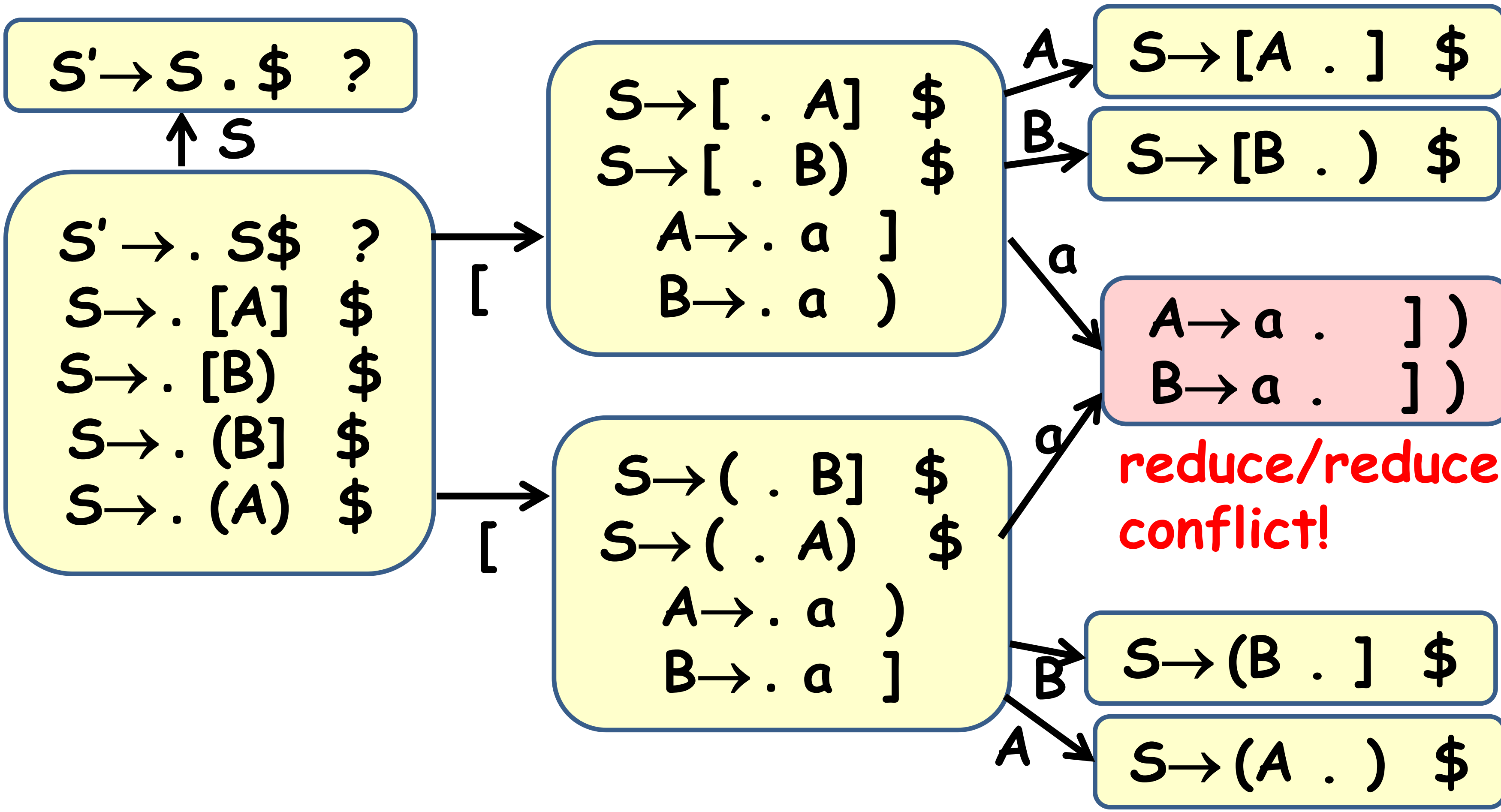
LR(1)で扱えてLALR(1)で扱えない文法

{S'→S$, S→[A] | [B) | (B] | (A),
A →a, B →a}

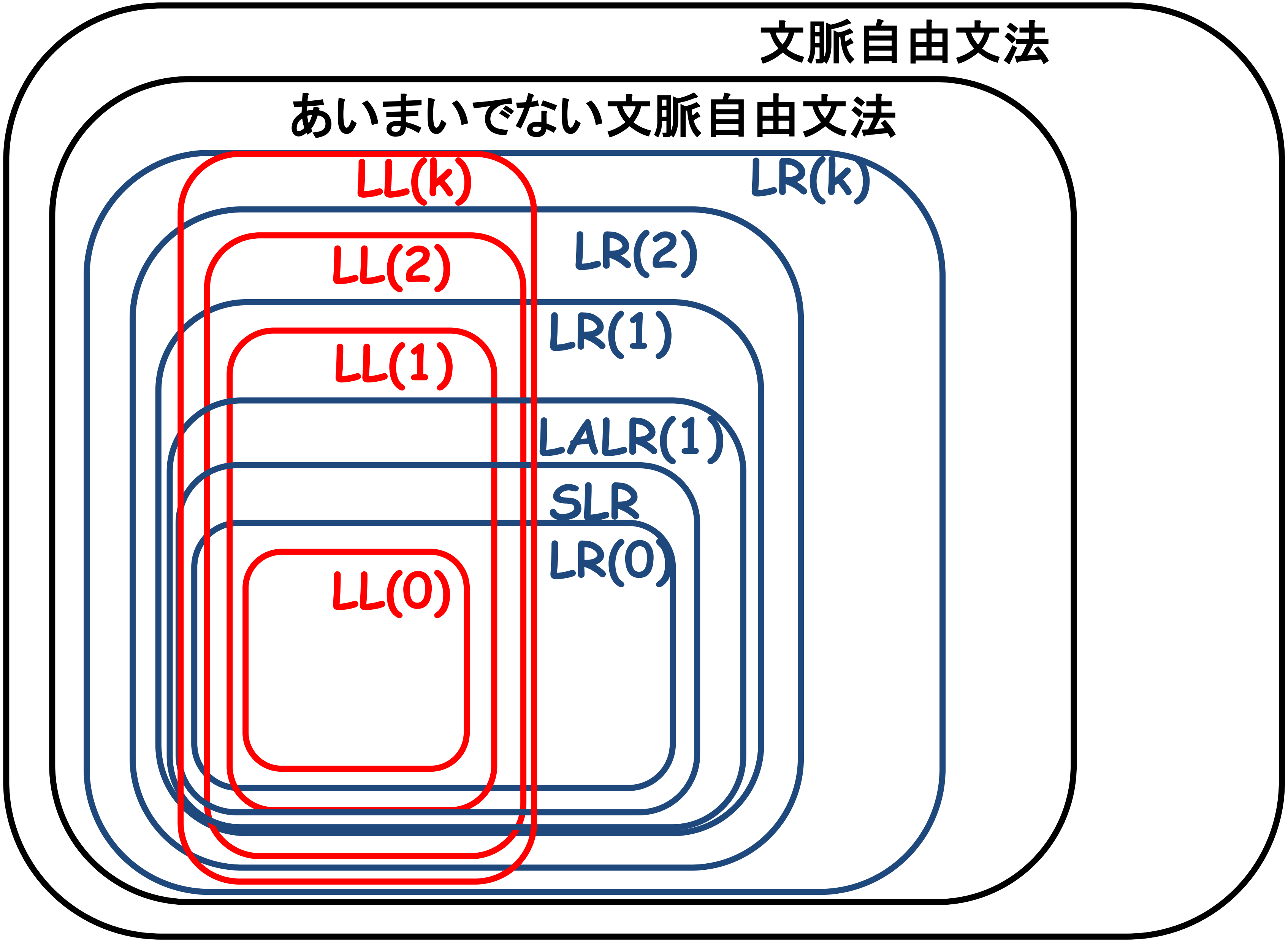


LR(1)で扱えてLALR(1)で扱えない文法

{S'→S$, S→[A] | [B) | (B] | (A),
A →a, B →a}

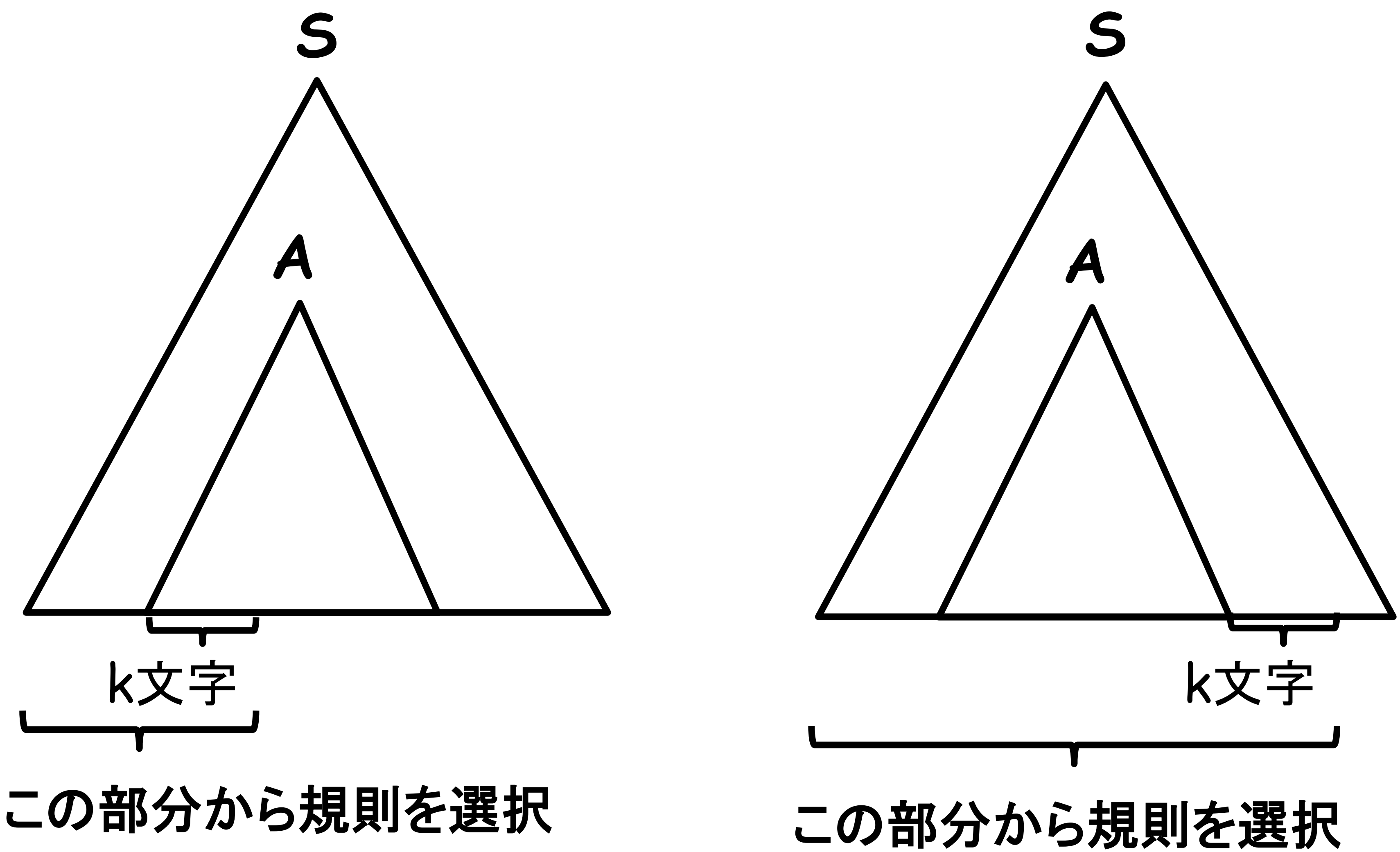


文法クラスの分類

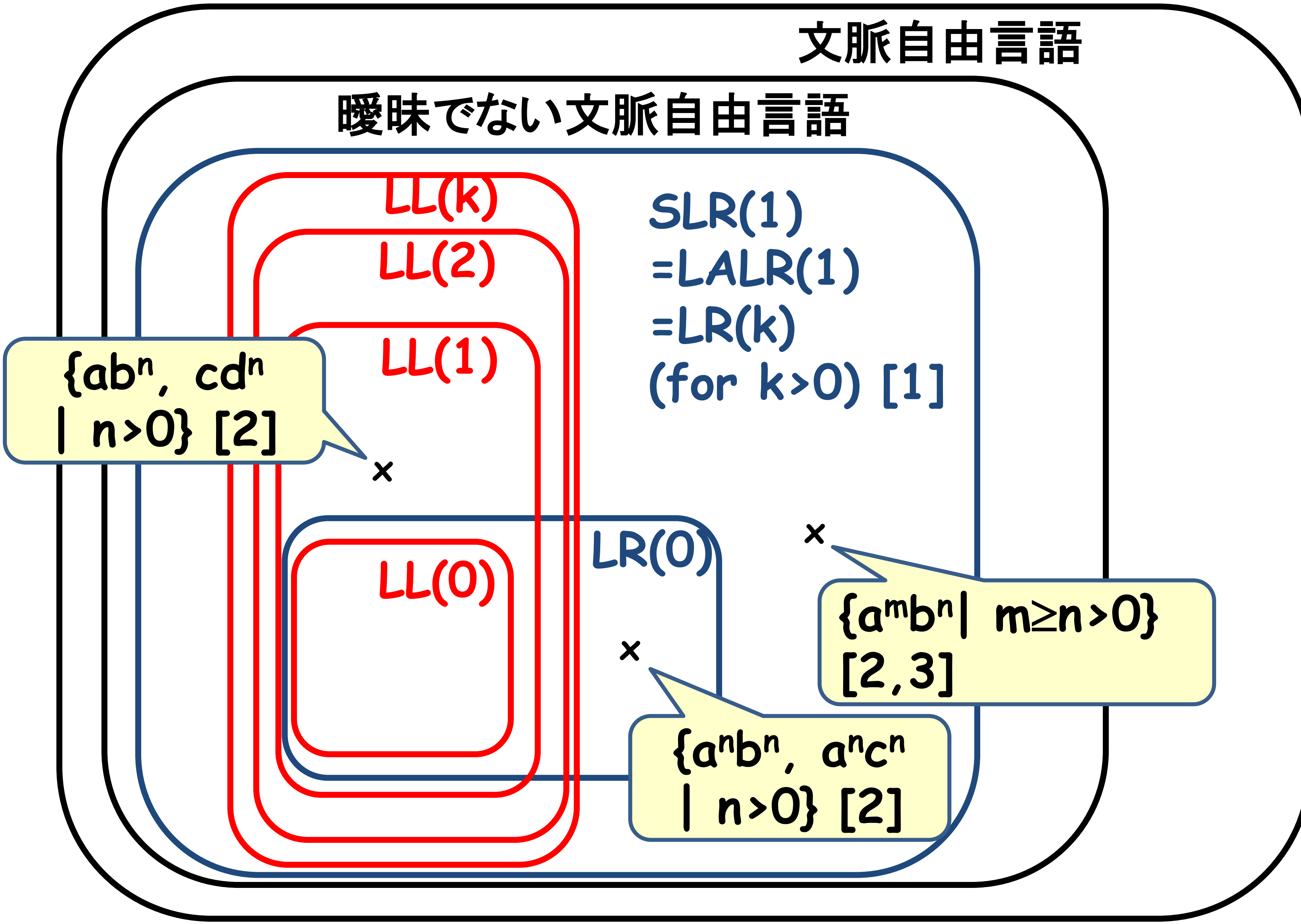


文脈自由文法
あいまいでない文脈自由文法
LL(k)
LR(k)
LL(2)
LR(2)
LR(1)
LL(1)
LALR(1)
SLR
LR(0)
LL(0)

LL(k) vs LR(k)



言語クラスの分類(参考)


文脈自由言語
曖昧でない文脈自由言語
LL(k)
LL(2)
LL(1)
LL(0)
SLR(1)
=LALR(1)
=LR(k)
(for k>0) [1]
LR(0)
{ab^n, cd^n
| n>0} [2]
{a^m b^n | m≥n>0}
[2,3]
{a^n b^n, a^n c^n
| n>0} [2]

言語クラスの分類に関する参考文献

[1] Mickunas, Lancaster, and Schneider,
"Transforming LR(k) Grammars to LR(1), SLR(1) and
(1,1) Bounded Right Context Grammars", JACM, 23(3), 1976.
(SLR(1)=LR(k) for all k>0の証明)

[2] Beatty, "Two Iteration Theorems for the LL(k) Languages",
Theoretical Computer Science, 12, 1980
($\{a^n b^n, a^n c^n \mid n>0\}$ や$\{a^m b^n \mid m \geq n>0\}$ がLL(k)でないことの証明)

[3] Geller and Harrison, "On LR(k) Grammars and Languages",
Theoretical Computer Science, 4, 1977
(LR(0)言語を特徴づけるTheorem 3.1(b)から$\{a^m b^n \mid m \geq n>0\}$ がLR(0)
でないことが導ける)

[4] Rosenkrantz and Stearns, "Properties of Deterministic Top
Down Grammars", Information and Control, 17(3), 1970
(LL(k)言語の階層がstrictであることの証明)

レポート課題

- 教科書 Exercise 3.12 (p.85)