Suponha que $G = (V, \Sigma, R, S)$ é uma gramática na Forma Normal de Chomsky (F.N.C).

Então as regras de G são da forma

$$\boxed{A \to \sigma}$$

(para algum $\sigma \in \Sigma$) ou

$$\boxed{A \to BC}$$

para $B, C \in V \setminus \{S\}$, com excessão da regra $S \to \varepsilon$ que pode ou não estar em R.

<u>PROBLEMA</u>: Dada $w \in \Sigma^*$, decidir se $w \in L(G)$, ou seja, decidir se

$$S \xRightarrow{*} w.$$

Vamos primeiro estudar um algoritmo recursivo para resolver uma versão um pouco mais geral desse problema:

<u>PROBLEMA</u>: Dada $A \in V$ e $w \in \Sigma^*$, decidir se $A \xRightarrow{*} w$.

A idéia principal do algoritmo vem do seguinte fato:

Se $B \overset{*}{\Rightarrow} x$ e $C \overset{*}{\Rightarrow} y$ e $A \to BC$ é uma regra da gramática, então

$$A \Rightarrow BC \overset{*}{\Rightarrow} xC \overset{*}{\Rightarrow} xy,$$

ou seja, então

$$A \overset{*}{\Rightarrow} xy.$$

Assim, se estamos tentando determinar se $A \overset{*}{\Rightarrow} w$, podemos quebrar w em 2 pedaços x e y e, para todas as regras $A \to BC$, decidir (recursivamente) se $B \overset{*}{\Rightarrow} x$ e se $C \overset{*}{\Rightarrow} y$.

O algoritmo recursivo, então, fica assim:

▷ supõe que $G = (V, \Sigma, R, S)$ é uma "variável global".

▷ supõe que $w = \sigma_1 \sigma_2 \cdots \sigma_n$

P.R. $(A, w[i..j])$

▷ devolve true se $A \overset{*}{\Rightarrow} \sigma_i \sigma_{i+1} \cdots \sigma_j$ e false caso contrário.

```
se j < i então                         (w[i..j] = ε)
    se A → ε ∈ R então
        devolve true
    senão
        devolve false
se i = j então                         (w = σ_i = σ_j)
    se A → σ_i ∈ R então
        devolve true
    senão
        devolve false
```

▷ agora os casos não básicos:

```
para k de i até j-1 faça
    para toda regra A → BC ∈ R faça
        se PR(B, w[i..k]) e PR(C, w[k+1..j])
            devolve true
devolve false
```

Note que este algoritmo faz <u>muitas</u> chamadas recursivas.

Por isso ele pode ser extremamente ineficiente, principalmente se a palavra w não estiver na linguagem descrita pela gramática.

<u>Observação</u>: note que, se a palavra do argumento é uma subpalavra de w ($w[i..j]$ é uma subpalavra de w), então todas as chamadas recursivas também possuem uma subpalavra de w no argumento (para todo $k \in \{i, \dots, j-1\}$ tanto $w[i..k]$ como $w[k+1..j]$ são subpalavras de w).

Portanto poderíamos fazer com que

$w = \sigma_1 \sigma_2 \dots \sigma_n$ fosse uma "variável global"

e simplesmente passar os índices do início e final da subpalavra.

O novo cabeçalho da função recursiva ficaria assim:

P.R. (A, i, j)

e as chamadas recursivas da linha 13, assim:

<u>se</u> $PR(B, i, k)$ e $PR(C, k+1, j)$ <u>então</u>

Mas isso não altera em nada a ineficiência da função recursiva...

...só nos ajuda a perceber que o número de chamadas recursivas realmente distintas é limitado por:

$$|V| \times \underbrace{(\text{número de subpalavras de } w)}_{|sub(w)|}.$$

$$|sub(w)| = \sum_{i=1}^{n} \sum_{j=i}^{n} 1 = \sum_{i=1}^{n} (n-i+1) = \sum_{k=1}^{n} k = \binom{n+1}{2}.$$

Portanto $\#$ chamadas distintas $= |V| \cdot \binom{n+1}{2} \approx |V| \frac{n^2}{2}$.

$= O(n^2)$.

Algoritmo recursivo com memorização das respostas (memoization)

▷ Usa matriz M $|V| \times n \times n$ <u>inicialmente</u> <u>vazia</u>

$M[A][i][j]$ = guarda se é possível que $A \overset{*}{\Rightarrow} \omega[i..j]$.

```
PR2(A, i, j)
  se M[A][i][j] ≠ Ø então devolve M[A][i][j].
  se j < i então devolve A → ε ∈ R
  se i = j então devolve A → σ_i ∈ R
  para k de i até j-1 faça
      para toda regra da forma A → BC faça
          se PR2(B, i, k) e PR2(C, k+1, j) então
              M[A][i][j] ← true
              devolve true
  M[A][i][j] ← false
  devolve false
```

(Neste pseudo código $\varnothing \neq$ false)

Algoritmo de Cocke-Younger-Kasami (CKY ou CYK)

- Baseado no algoritmo recursivo que vimos.
- Usa programação dinâmica (portanto sem recursão).

Matriz auxiliar S $n \times n$

$$S_{ij} = \{ A \in V : A \overset{*}{\Rightarrow} \sigma_i \cdots \sigma_j \},$$

ou seja, S_{ij} é o conjunto das variáveis capazes de gerar a subpalavra $w[i \ldots j]$.

[Discussão na lousa p/ descobrir como preencher S.]

Inicialização:

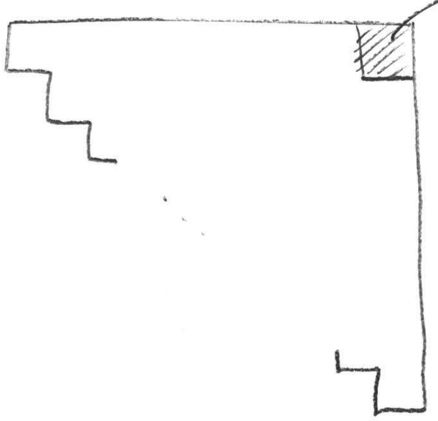
- todas as entradas de S começam vazias
- S_{ii} tem as variáveis que conseguem gerar σ_i, ou seja,

$$S_{ii} = \{ A \in V : A \rightarrow \sigma_i \in R \}.$$

CYK $(G = (V, \Sigma, R, S)\ ,\ w = \sigma_1 \sigma_2 \cdots \sigma_n)$

para i **de** 1 **até** n **faça**
    para j **de** i **até** n **faça**
        $S_{ij} \leftarrow \emptyset$
    para **toda** regra $A \to \sigma_i$ **faça**
        $S_{ii} \leftarrow S_{ii} \cup \{A\}$
para l **de** 2 **até** n **faça**
    para i **de** 1 **até** $n - l + 1$ **faça**
        $j \leftarrow i + l - 1$
        para k **de** i **até** $j-1$ **faça**
            para **toda** regra $A \to BC$ **faça**
                se $B \in S_{ik}$ e $C \in S_{(k+1)j}$ **então**
                    $S_{ij} \leftarrow S_{ij} \cup \{A\}$
devolva S

PERGUNTA: Qual a posição de interesse da matriz S?



S_{1n} = conjunto das variáveis que geram $\underbrace{\sigma_1 \cdots \sigma_n}_{w}$

$S \in S_{1n} \Rightarrow w \in L(G)$

$S \notin S_{1n} \Rightarrow w \notin L(G)$.

$S \to a \mid c \mid \varepsilon$

$A \to a$

$Z \to a$

$B \to b$

$Y \to c$

$C \to c$

$S \to XY \mid ZW \mid BR \mid BC \mid AZ \mid AT \mid AB \mid YC$

$X \to AT \mid AB$

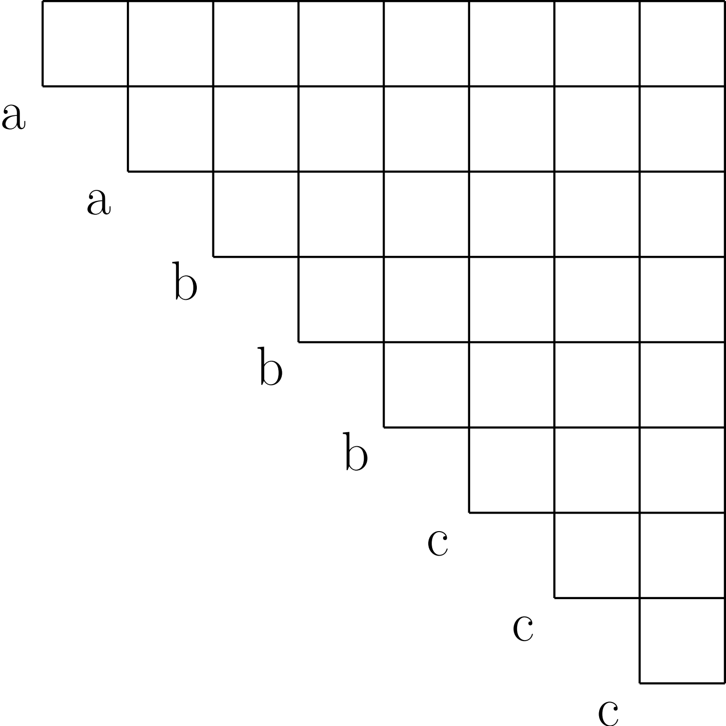
$Y \to YC$

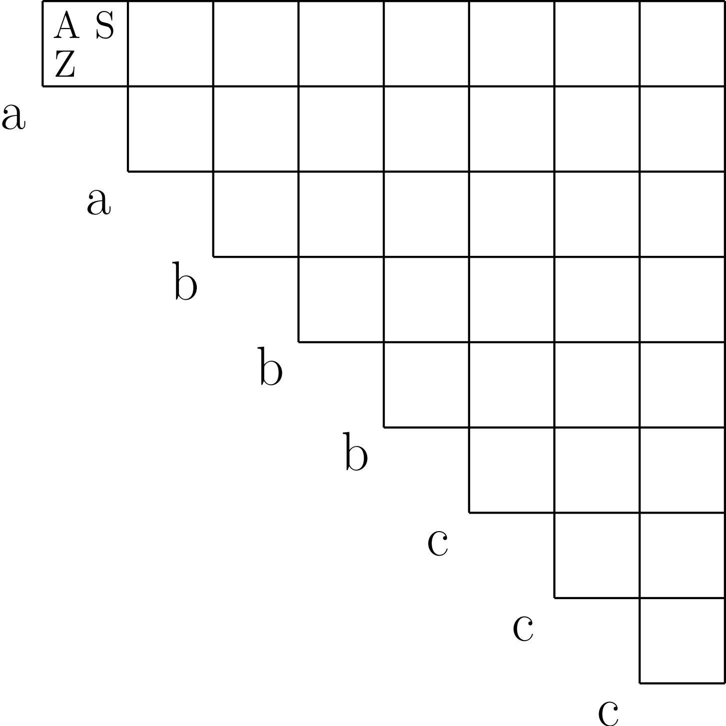
$Z \to AZ$

$W \to BR \mid BC$

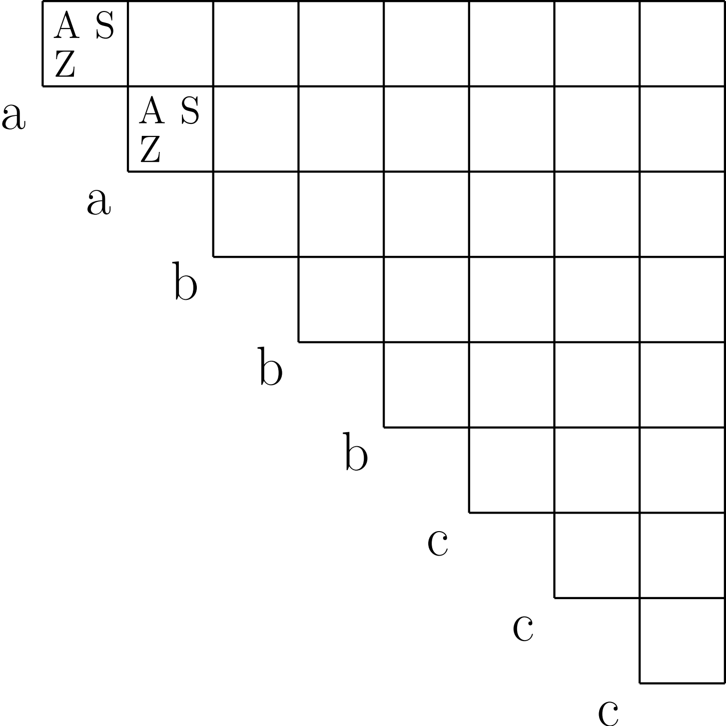
$R \to WC$

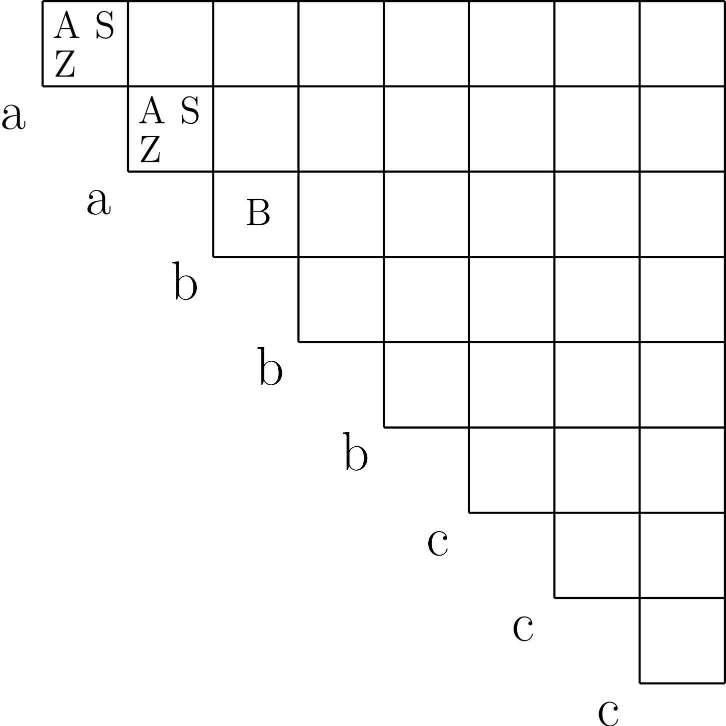
$T \to XB$

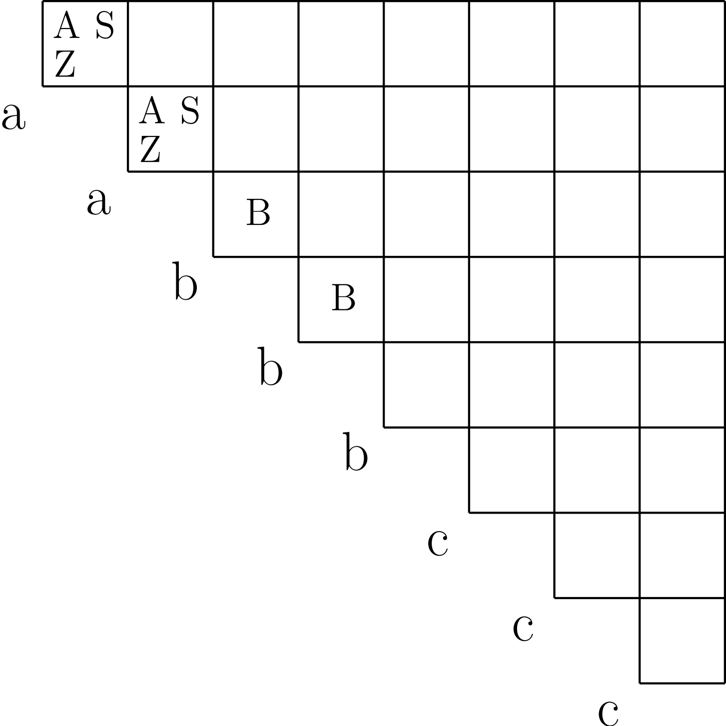

a
a
b
b
b
c
c
c

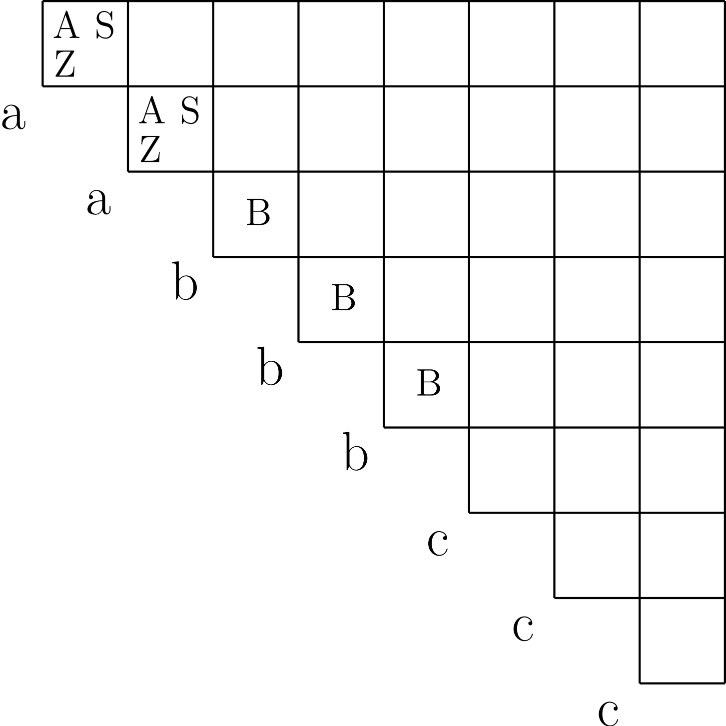


A S
Z
a
a
b
b
b
c
c
c

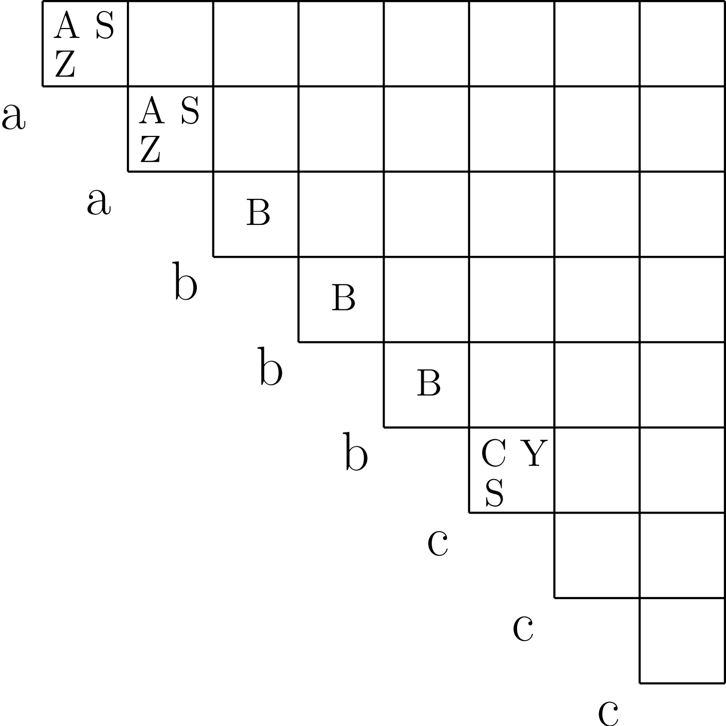

A S
Z
A S
Z
a
a
b
b
b
c
c
c

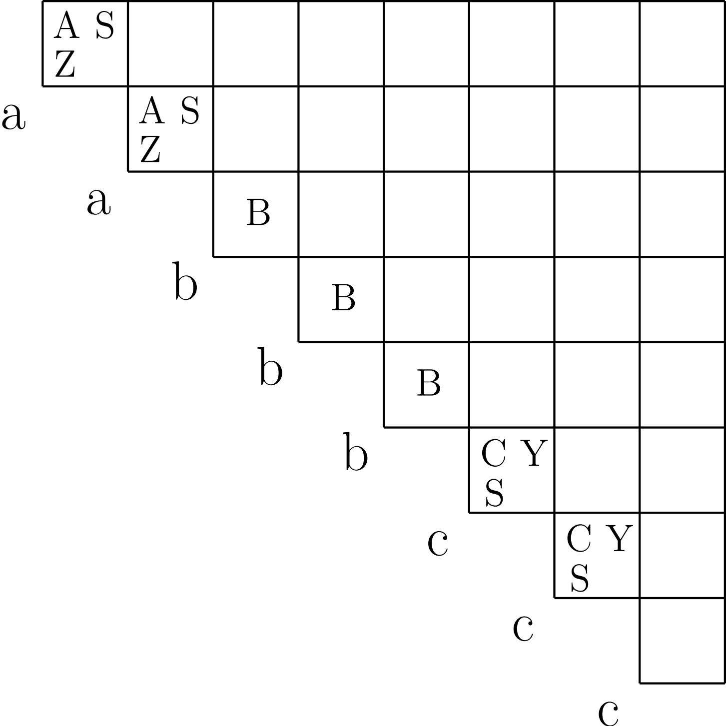

A S
Z
a
A S
Z
a
B
b
b
b
c
c
c


A S Z
A S Z
B
B
a
a
b
b
b
c
c
c



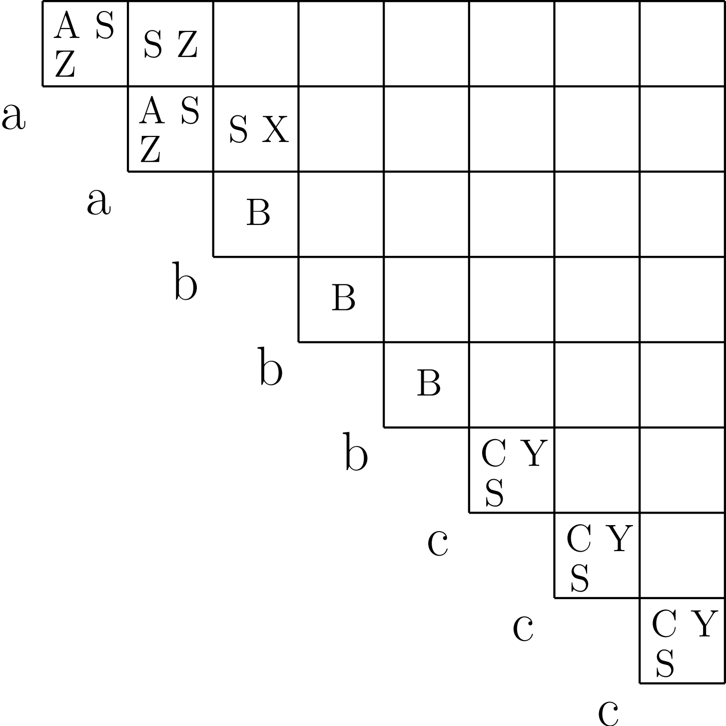
A S
Z
A S
Z
B
B
B
a
a
b
b
b
c
c
c


A S
Z
a
A S
Z
a
B
b
B
b
B
b
C Y
S
c
c
c


A S Z
A S Z
B
B
B
C Y S
C Y S
a
a
b
b
b
c
c
c

A S Z							
	A S Z						
		B					
			B				
				B			
					C Y S		
						C Y S	
							C Y S
a	a	b	b	b	c	c	c

A S Z	S Z						
	A S Z						
		B					
			B				
				B			
					C Y S		
						C Y S	
							C Y S
a	a	b	b	b	c	c	c


A S Z
S Z
A S Z
S X
B
B
B
C Y S
C Y S
C Y S
a
a
b
b
b
c
c
c

A S Z	S Z						
a	A S Z	S X					
	a	B	∅				
		b	B				
			b	B			
				b	C Y S		
					c	C Y S	
						c	C Y S
							c

A S Z	S Z						
	A S Z	S X					
		B	∅				
			B	∅			
				B			
					C Y S		
						C Y S	
							C Y S
a	a	b	b	b	c	c	c

	A S Z	S Z						
a		A S Z	S X					
	a		B	$\emptyset$				
		b		B	$\emptyset$			
			b		B	S W		
				b		C Y S		
					c		C Y S	
						c		C Y S
							c	

a	A S Z	S Z						
a		A S Z	S X					
b			B	∅				
b				B	∅			
b					B	S W		
c						C Y S	S Y	
c							C Y S	
c								C Y S

a	A S Z	S Z						
a		A S Z	S X					
b			B	∅				
b				B	∅			
b					B	S W		
c						C Y S	S Y	
c							C Y S	S Y
c								C Y S

A S Z	S Z	$\emptyset$					
	A S Z	S X					
		B	$\emptyset$				
			B	$\emptyset$			
				B	S W		
					C Y S	S Y	
						C Y S	S Y
							C Y S
a	a	b	b	b	c	c	c

	A S Z	S Z	∅					
a		A S Z	S X	T				
	a		B	∅				
		b		B	∅			
			b		B	S W		
				b		C Y S	S Y	
					c		C Y S	S Y
						c		C Y S
							c	

A S Z	S Z	∅					
a	A S Z	S X	T				
	a	B	∅	∅			
		b	B	∅			
			b	B	S W		
				b	C Y S	S Y	
					c	C Y S	S Y
						c	C Y S
							c

	A S Z	S Z	∅					
a		A S Z	S X	T				
	a		B	∅	∅			
		b		B	∅	∅		
			b		B	S W		
				b		C Y S	S Y	
					c		C Y S	S Y
						c		C Y S
							c	

	A S Z	S Z	∅					
a		A S Z	S X	T				
	a		B	∅	∅			
		b		B	∅	∅		
			b		B	S W	R	
				b		C Y S	S Y	
					c		C Y S	S Y
						c		C Y S
							c	

a	A S Z	S Z	∅					
a		A S Z	S X	T				
b			B	∅	∅			
b				B	∅	∅		
b					B	S W	R	
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

a	A S Z	S Z	∅	S X				
a		A S Z	S X	T				
b			B	∅	∅			
b				B	∅	∅		
b					B	S W	R	
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

	A S Z	S Z	∅	S X				
a		A S Z	S X	T	∅			
	a		B	∅	∅			
		b		B	∅	∅		
			b		B	S W	R	
				b		C Y S	S Y	S Y
					c		C Y S	S Y
						c		C Y S
							c	

a	A S Z	S Z	$\emptyset$	S X				
a		A S Z	S X	T	$\emptyset$			
b			B	$\emptyset$	$\emptyset$	$\emptyset$		
b				B	$\emptyset$	$\emptyset$		
b					B	S W	R	
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

A S Z	S Z	$\emptyset$	S X				
a	A S Z	S X	T	$\emptyset$			
	a	B	$\emptyset$	$\emptyset$	$\emptyset$		
		b	B	$\emptyset$	$\emptyset$	S W	
			b	B	S W	R	
				b	C Y S	S Y	S Y
					c	C Y S	S Y
						c	C Y S
							c

	A S Z	S Z	∅	S X				
a		A S Z	S X	T	∅			
	a		B	∅	∅	∅		
		b		B	∅	∅	S W	
			b		B	S W	R	∅
				b		C Y S	S Y	S Y
					c		C Y S	S Y
						c		C Y S
							c	

A S Z	S Z	$\emptyset$	S X	T			
a	A S Z	S X	T	$\emptyset$			
	a	B	$\emptyset$	$\emptyset$	$\emptyset$		
		b	B	$\emptyset$	$\emptyset$	S W	
			b	B	S W	R	$\emptyset$
				b	C Y S	S Y	S Y
					c	C Y S	S Y
						c	C Y S
							c

a	A S Z	S Z	∅	S X	T			
a		A S Z	S X	T	∅	∅		
b			B	∅	∅	∅		
b				B	∅	∅	S W	
b					B	S W	R	∅
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

A S Z	S Z	∅	S X	T			
a	A S Z	S X	T	∅	∅		
	a	B	∅	∅	∅	∅	
		b	B	∅	∅	S W	
			b	B	S W	R	∅
				b	C Y S	S Y	S Y
					c	C Y S	S Y
						c	C Y S
							c

A S Z	S Z	$\emptyset$	S X	T			
a	A S Z	S X	T	$\emptyset$	$\emptyset$		
	a	B	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	
		b	B	$\emptyset$	$\emptyset$	S W	R
			b	B	S W	R	$\emptyset$
				b	C Y S	S Y	S Y
					c	C Y S	S Y
						c	C Y S
							c

a	A S Z	S Z	∅	S X	T	∅		
a		A S Z	S X	T	∅	∅		
b			B	∅	∅	∅	∅	
b				B	∅	∅	S W	R
b					B	S W	R	∅
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

a	A S Z	S Z	∅	S X	T	∅		
a		A S Z	S X	T	∅	∅	∅	
b			B	∅	∅	∅	∅	
b				B	∅	∅	S W	R
b					B	S W	R	∅
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

A S Z	S Z	∅	S X	T	∅		
a	A S Z	S X	T	∅	∅	∅	
	a	B	∅	∅	∅	∅	S W
		b	B	∅	∅	S W	R
			b	B	S W	R	∅
				b	C Y S	S Y	S Y
					c	C Y S	S Y
						c	C Y S
							c

a	A S Z	S Z	∅	S X	T	∅	∅	
a		A S Z	S X	T	∅	∅	∅	
b			B	∅	∅	∅	∅	S W
b				B	∅	∅	S W	R
b					B	S W	R	∅
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

a	A S Z	S Z	∅	S X	T	∅	∅	
a		A S Z	S X	T	∅	∅	∅	S
b			B	∅	∅	∅	∅	S W
b				B	∅	∅	S W	R
b					B	S W	R	∅
c						C Y S	S Y	S Y
c							C Y S	S Y
c								C Y S

	A S Z	S Z	∅	S X	T	∅	∅	S
a		A S Z	S X	T	∅	∅	∅	S
	a		B	∅	∅	∅	∅	S W
		b		B	∅	∅	S W	R
			b		B	S W	R	∅
				b		C Y S	S Y	S Y
					c		C Y S	S Y
						c		C Y S
							c	