Recuperação de Erros

PCS 3566/3866 – Linguagens e Compiladores

Aula 11

Prof. João José Neto

Recuperação de erros sintáticos

Recuperação de erros sintáticos

- **Não se trata de corrigir eventuais erros encontrados**
- **Refere-se à recondução do reconhecedor ou analisador sintático à condição de prosseguir sua atividade após a detecção de um erro de sintaxe na cadeia de entrada**

Estudam-se aqui quatro situações de recuperação de erros

1. **Em autômatos finitos**
2. **Em autômatos de pilha estruturados**
3. **Em analisadores determinísticos descendentes**
4. **Em analisadores determinísticos ascendentes**

1. RECUPERAÇÃO DE ERROS EM AUTÔMATOS FINITOS

Recuperação de Erros

- **Reconhecedores destinam-se a aceitar sentenças corretas da linguagem**
- **Os programadores dificilmente geram textos sintaticamente corretos**
- **Textos incorretos dessincronizam o reconhecedor**
- **Mecanismos de ressincronização permitem detectar mais de um erro no texto incorreto**
- **No caso geral a recuperação de erros é difícil**
- **A recuperação absoluta de erros é impossível, pois exigiria o conhecimento da sentença correta**

Erros simples

- **Correspondem a uma destas três ocorrências:**
  - ***omissão* de átomo s** — **asb → ab**
  - ***substituição* de um átomo s por outro x** — **asb → axb**
  - ***inserção* de um átomo x estranho** — **abc → axbc**
- **A grande maioria dos erros é simples**
- ***Erros múltiplos* são sobreposições de dois ou mais erros simples simultâneos**
- **Erros simples que não se superpõem não são considerados formadores de erros múltiplos**
- ***Manifestação da presença de erro* é a situação em que o autômato não consegue mais transitar, por efeito de um erro**

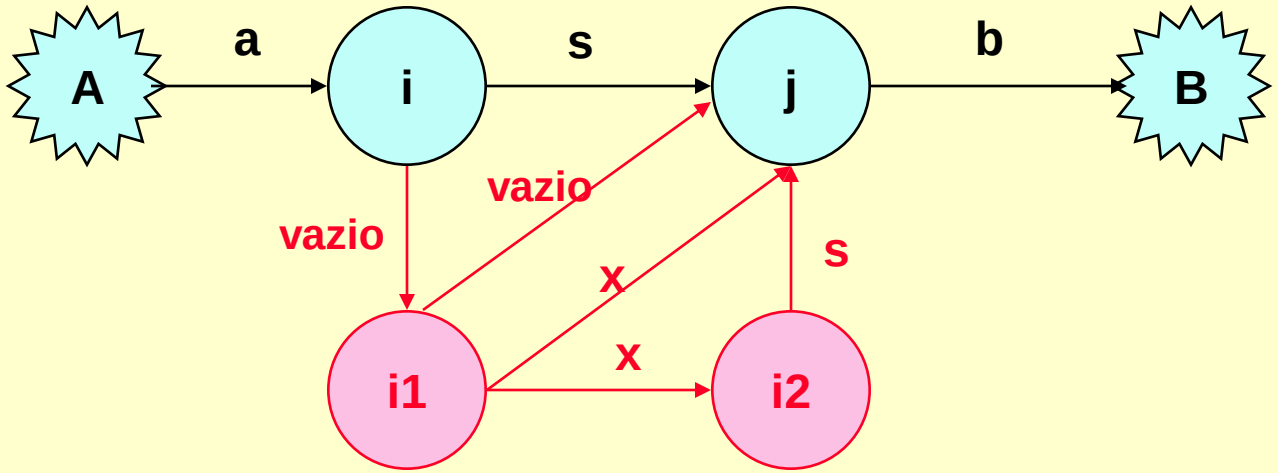
Observações

- **Nem sempre a manifestação da presença de um erro ocorre próxima do ponto de sua ocorrência real**
- ***Recuperação* não significa *correção* de erros, mas sim uma forma de ressincronização do autômato após a manifestação de um erro**
- **O uso da filosofia de correção automática é um artifício muito utilizado para possibilitar o prosseguimento da análise, e a localização de outros eventuais erros**

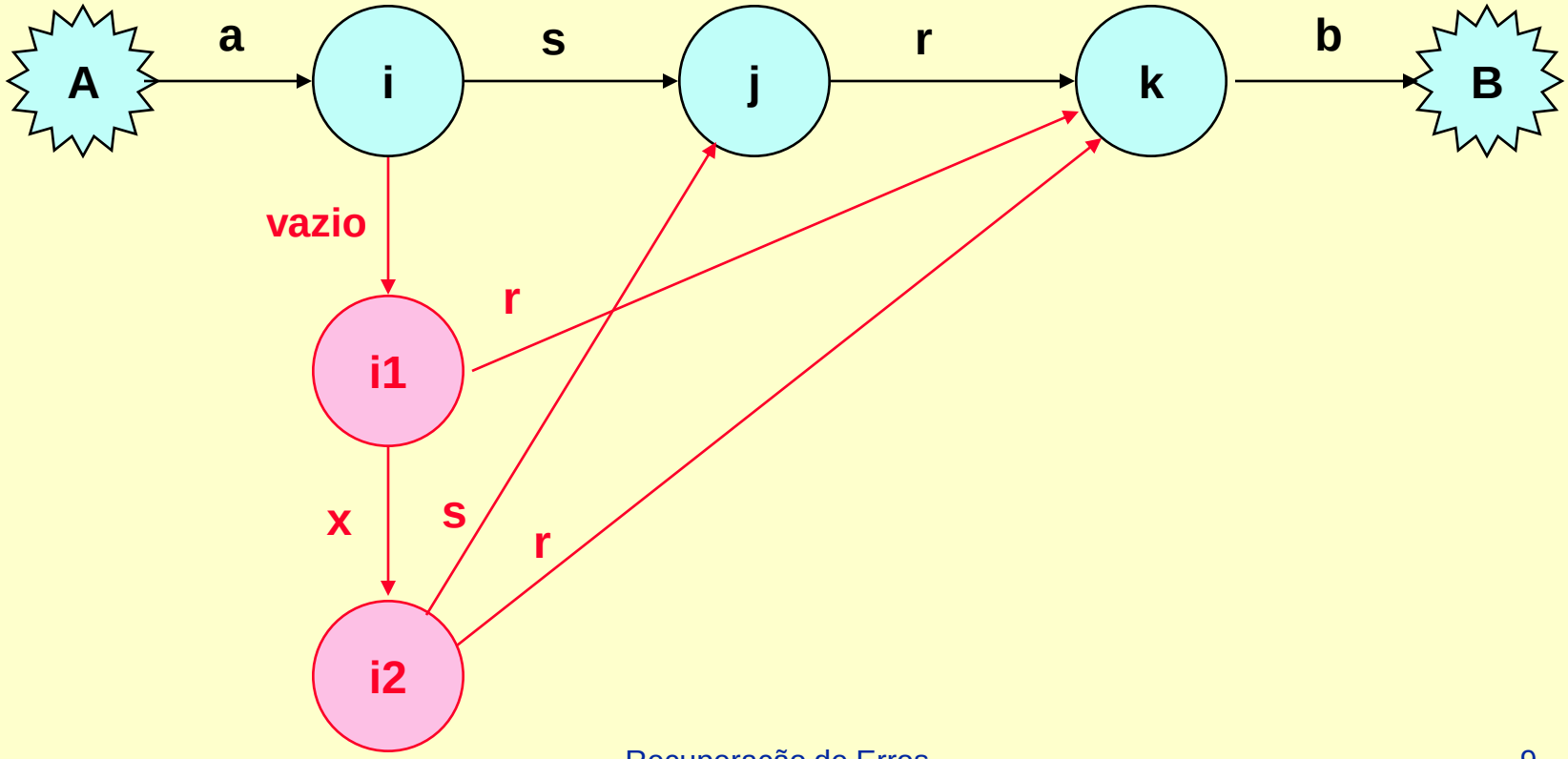
Recuperação de erros simples

- **Um símbolo s não permite transitar, impedindo o prosseguimento da análise**
- **Se o erro que se manifesta for simples, é possível recuperá-lo por correção:**
  - **Para o caso de inserção, elimina-se s**
  - **Para o caso de substituição, elimina-se s e insere-se x**
  - **Para o caso de omissão, insere-se x antes de s**
- **O texto a s b será alterado respectivamente para:**
  - **a b** **eliminando-se s**
  - **a x b** **trocando-se s por x**
  - **a x s b** **inserindo-se x antes de s**

- **Do ponto de vista do autômato, deverão ser aceitas as seguintes seqüências:**
  - **a s b**
  - **a b**
  - **a x b**
  - **a x s b**
- **Uma fusão conveniente dessas formas, mantendo em evidência a forma original a s b, isola-a das demais por uma transição em vazio:**



- **Atenuam-se os não-determinismos deste esquema explicitando-se os primeiros átomos r consumidos logo em seguida ao átomo que causou o erro.**
- **Mantém-se a transição em vazio que isola o autômato original do mecanismo de recuperação de erros simples.**

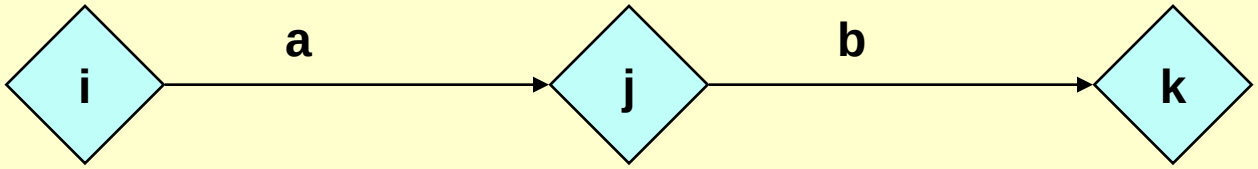


Recuperação de erros em autômatos finitos

- **A presença exclusiva de transições internas facilitam a recuperação de erros em autômatos finitos**
- **O tratamento de erros simples exige a análise separada dos estados internos e dos estados finais**
- **Erros múltiplos podem ser tratados por aplicações múltiplas do tratamento de erros simples ou pelo uso de heurísticas menos rigorosas**
- **Simplifica-se muito a recuperação de erros aplicando-a em versões determinísticas do autômato**

Recuperação de erros simples

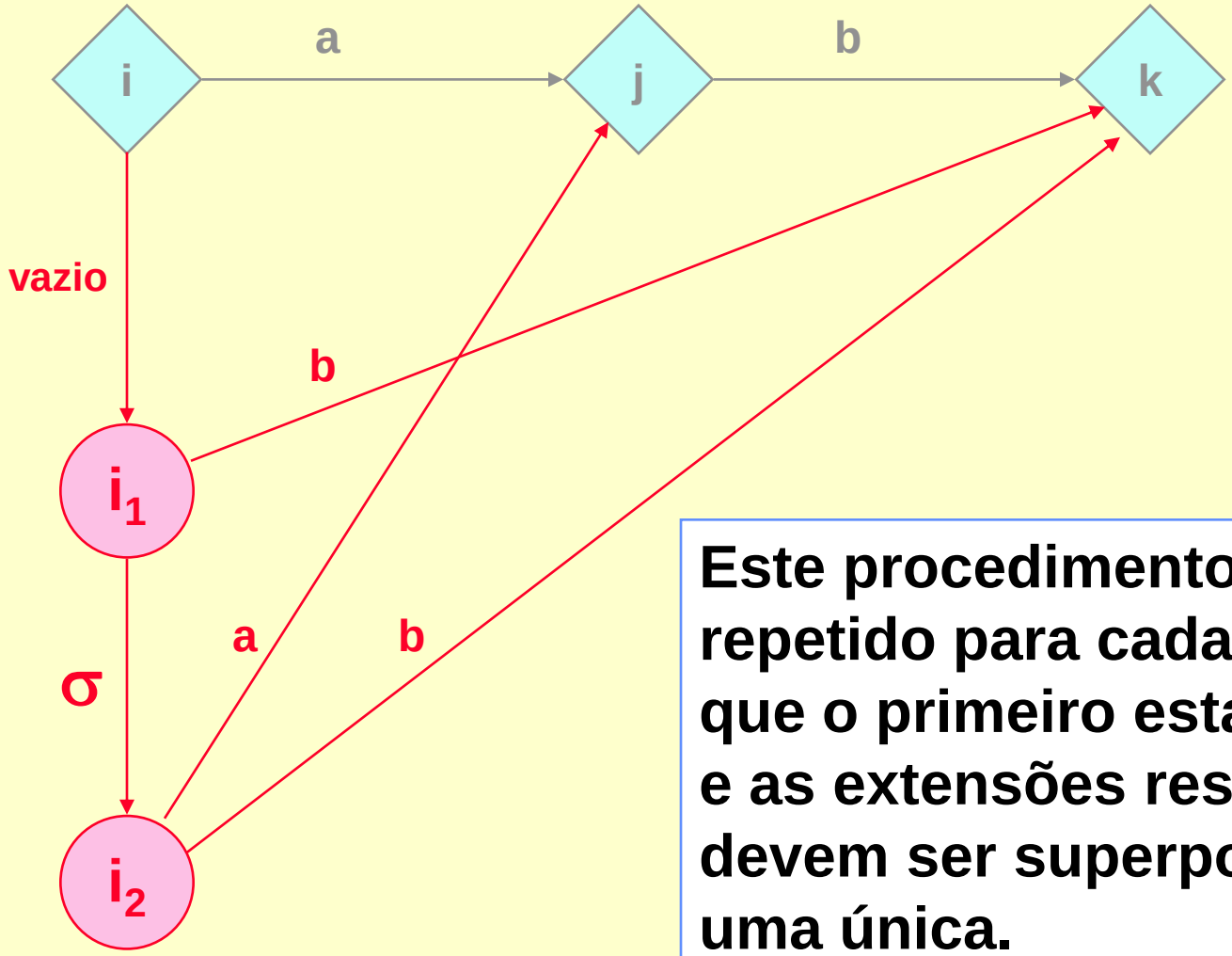
- **Cada estado i do autômato é um potencial ponto de manifestação de erro**
- **j é um dos estados *primeiros-sucessores* de i se houver alguma transição da forma (i , a) $\rightarrow$ j**
- **k é um dos estados *segundos-sucessores* de i se houver alguma transição do tipo (j , b) $\rightarrow$ k**
- **(i, j, k) representa uma seqüência de estados, não obrigatoriamente diferentes, denotada abaixo (não é um diagrama de estados!)**



Montagem do Autômato de Recuperação

- Para cada estado i do autômato, criar dois estados adicionais i_1 e i_2
- Conectar o estado i ao novo estado i_1 por meio de uma transição em vazio: $(i, \varepsilon) \rightarrow i_1$
- Incluir uma transição para a eliminação de átomos espúrios σ: $(i_1, \sigma) \rightarrow i_2$
- Incluir uma transição para recuperar a inserção de átomos σ: $(i_2, a) \rightarrow j$
- Incluir uma transição para recuperar a substituição de átomos a por σ: $(i_2, b) \rightarrow k$
- Incluir uma transição para recuperar a eliminação indevida de átomos a: $(i_1, b) \rightarrow k$

Aspecto da extensão de recuperação de erros simples para o estado i, relativa à tripla (i, j, k):

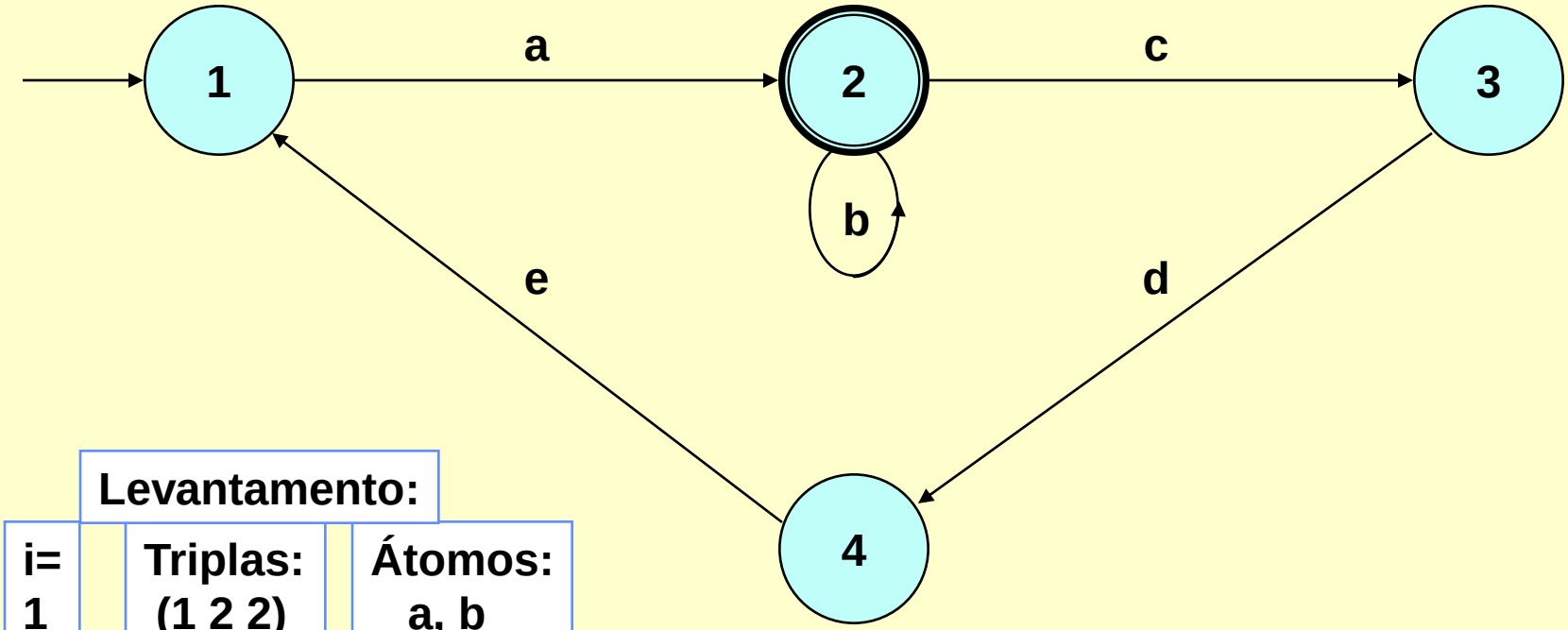


Este procedimento deve ser repetido para cada tripla em que o primeiro estado seja i e as extensões resultantes devem ser superpostas em uma única.

Observações

- **O método apresentado cobre apenas casos em que i não é estado final, e em que existem os estados primeiros-sucessores j e segundos-sucessores k para o estado i.**
- **Para i estado final, faz-se com que os estados i_1 e i_2 também sejam estados finais**
- **É irrelevante se k é ou não estado final**
- **Omitir transições para estados primeiros-sucessores ou segundos-sucessores que não existam**
- **Com esta complementação, o método permite criar extensões de recuperação absoluta de erros simples para todos os autômatos finitos determinísticos**

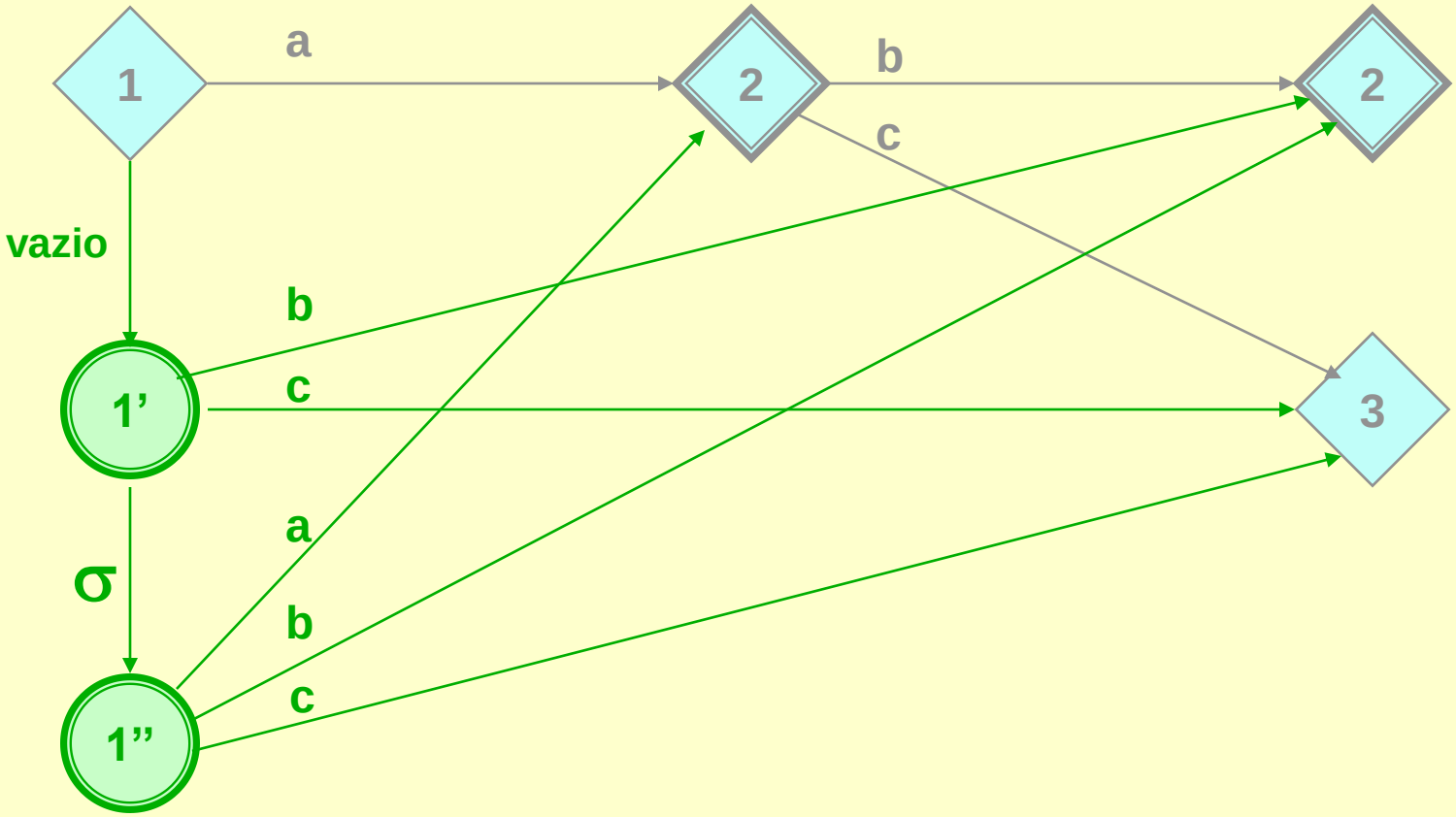
Exemplo



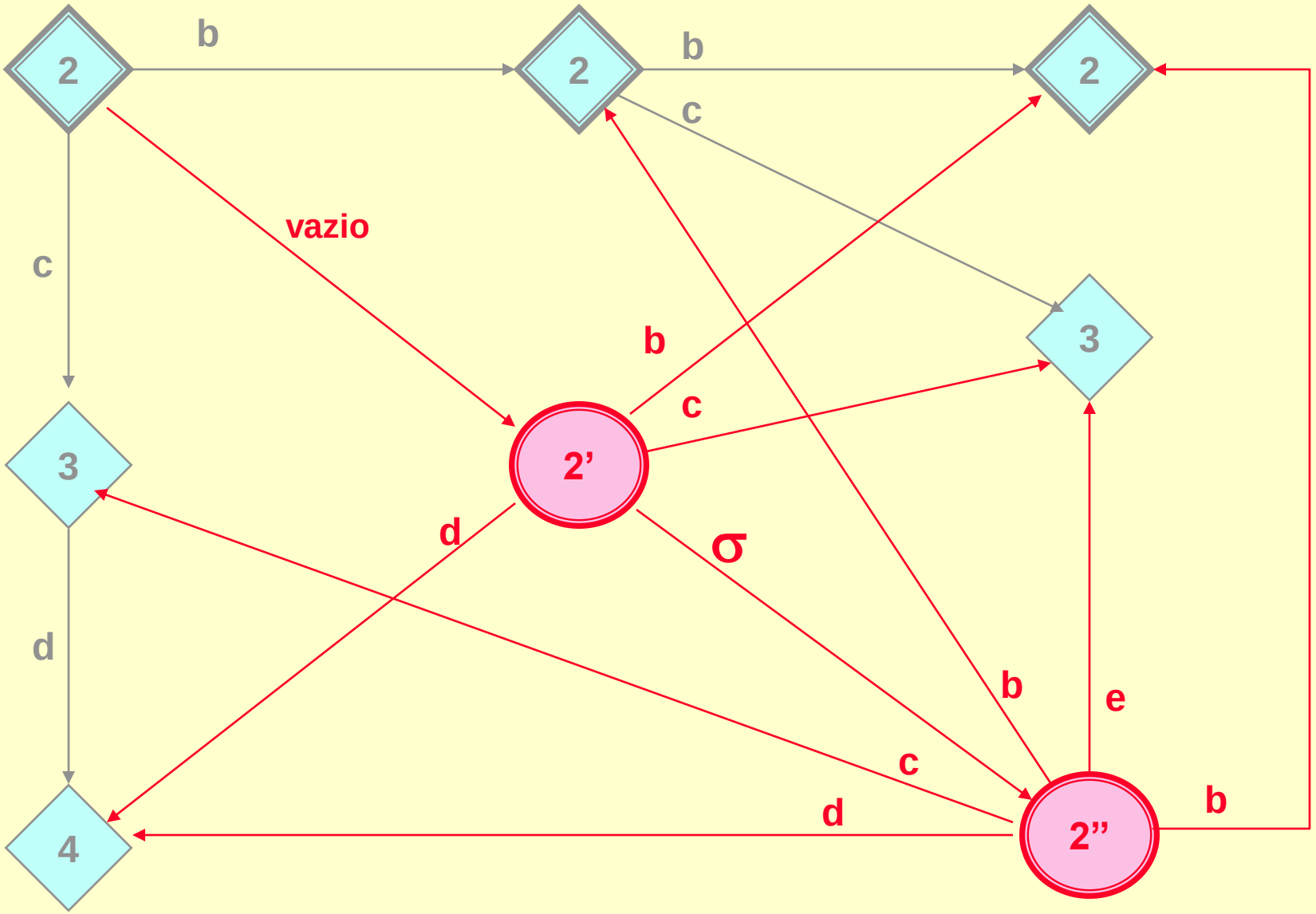
Levantamento:

i=	Triplas:	Átomos:
1	(1 2 2)	a, b
1	(1 2 3)	a, c
2	(2 2 2)	b, b
2	(2 2 3)	b, c
2	(2 3 4)	c, d
3	(3 4 1)	d, e
4	(4 1 2)	e, a

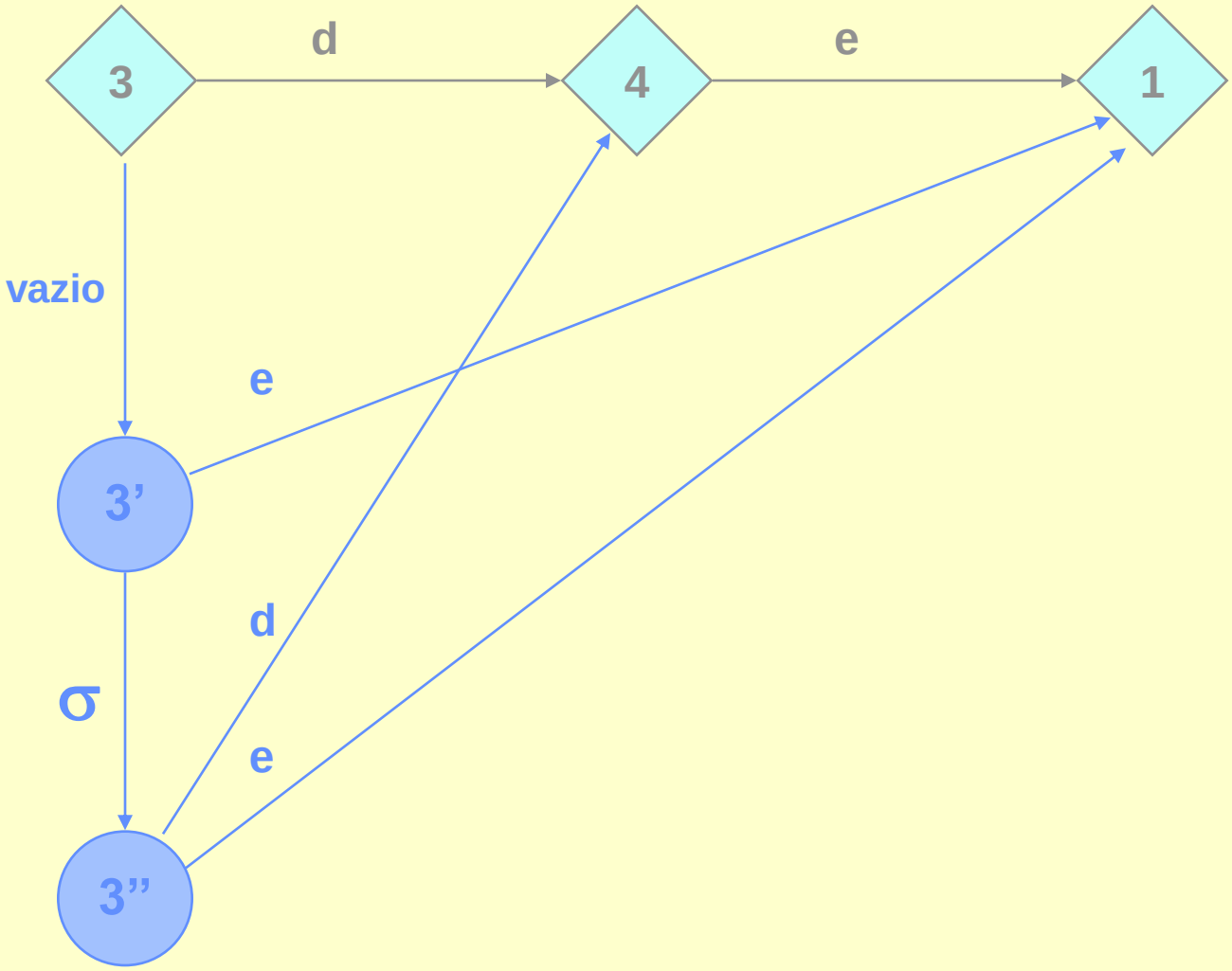
Recuperação no estado 1:



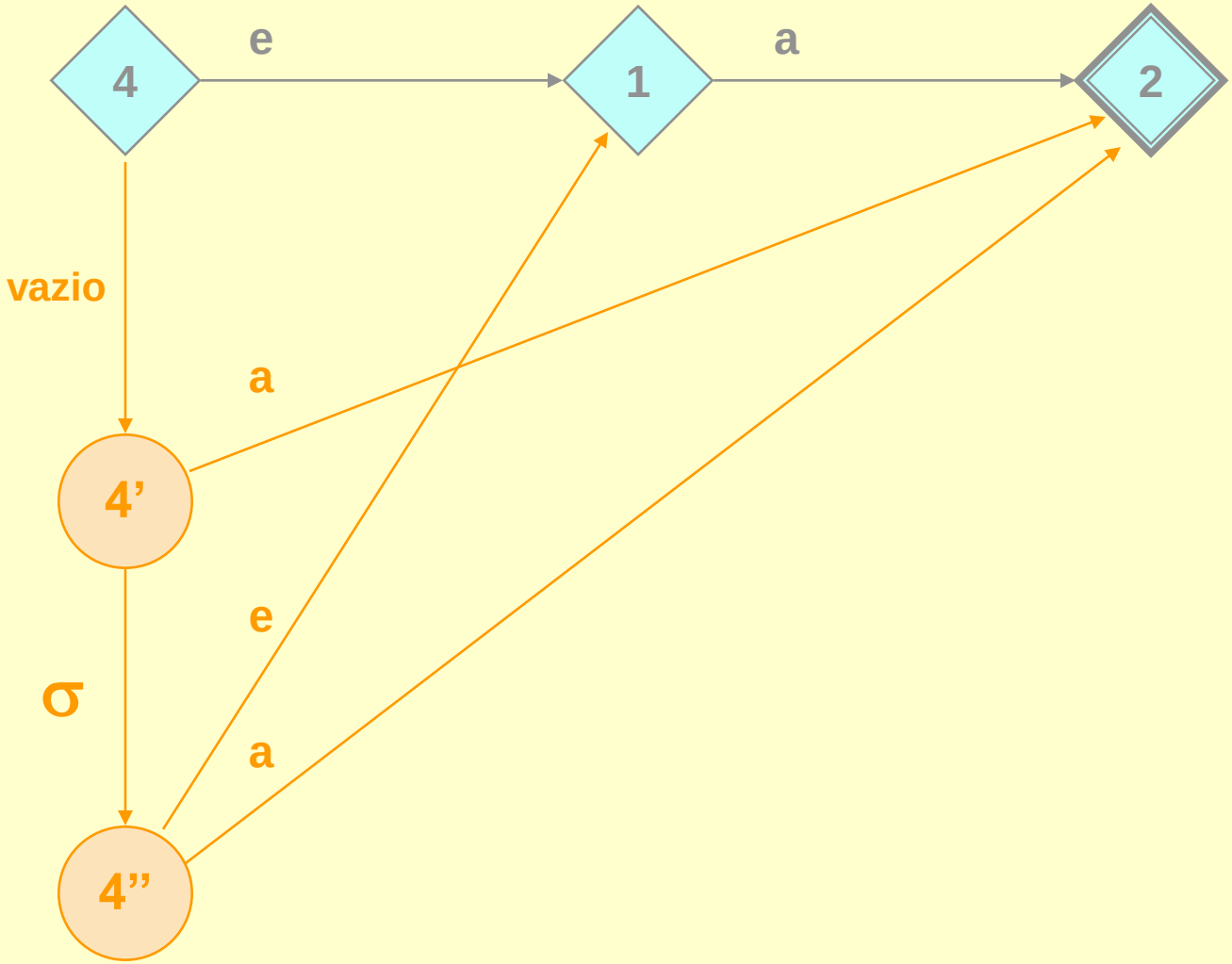
Recuperação no estado 2:

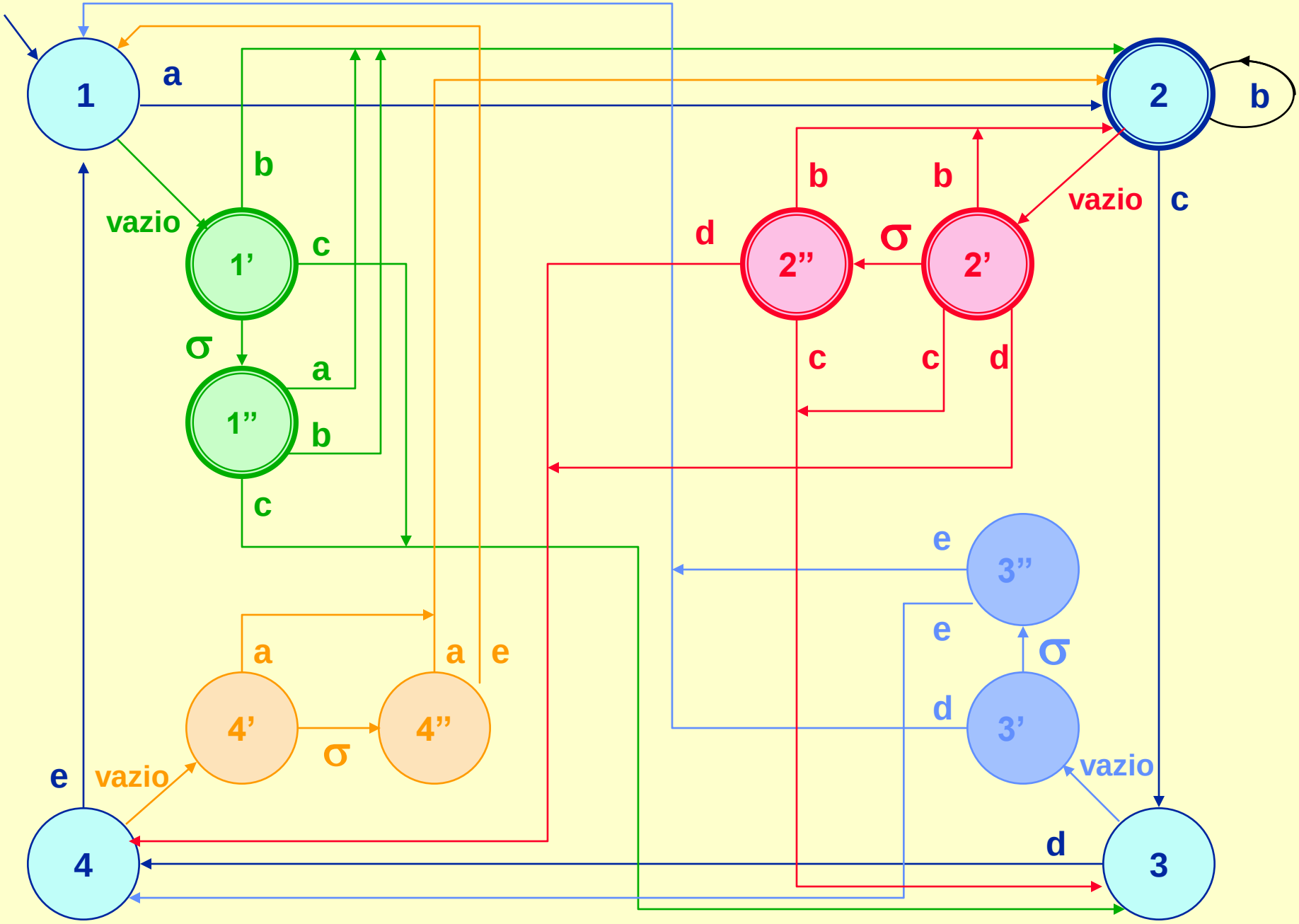


Recuperação no estado 3:



Recuperação no estado 4:





Forma final do autômato, com as extensões de recuperação de erros simples

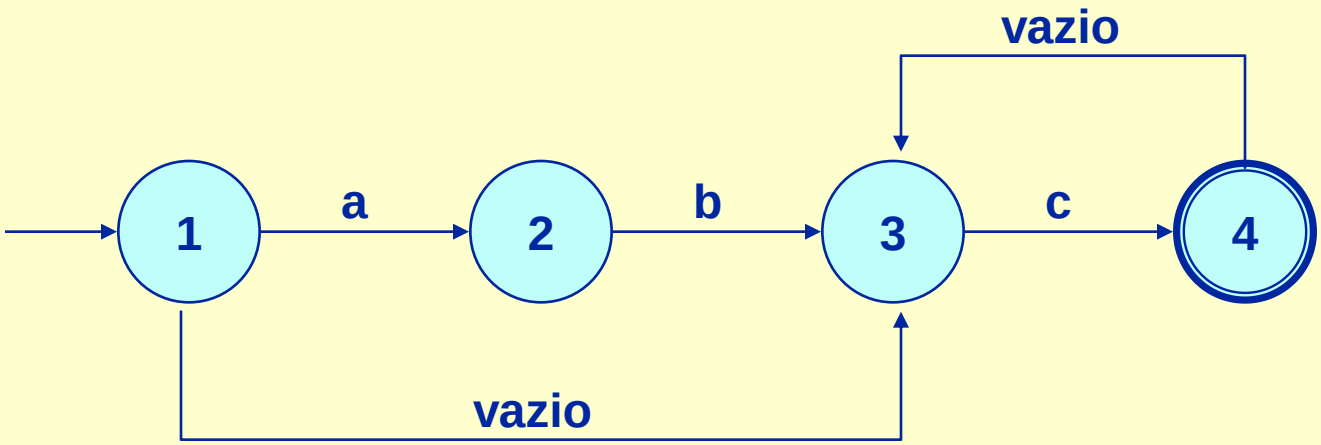
Observações

- **Notar que algumas das transições criadas se superpõem no autômato final**
- **Notar que, caso a, b, c, d, e não sejam todos distintos, é muito provável aparecerem não-determinismos**
- **Isto se deve à dificuldade de se determinar exatamente o erro realmente cometido**

Recuperação de erros em autômatos finitos não-determinísticos

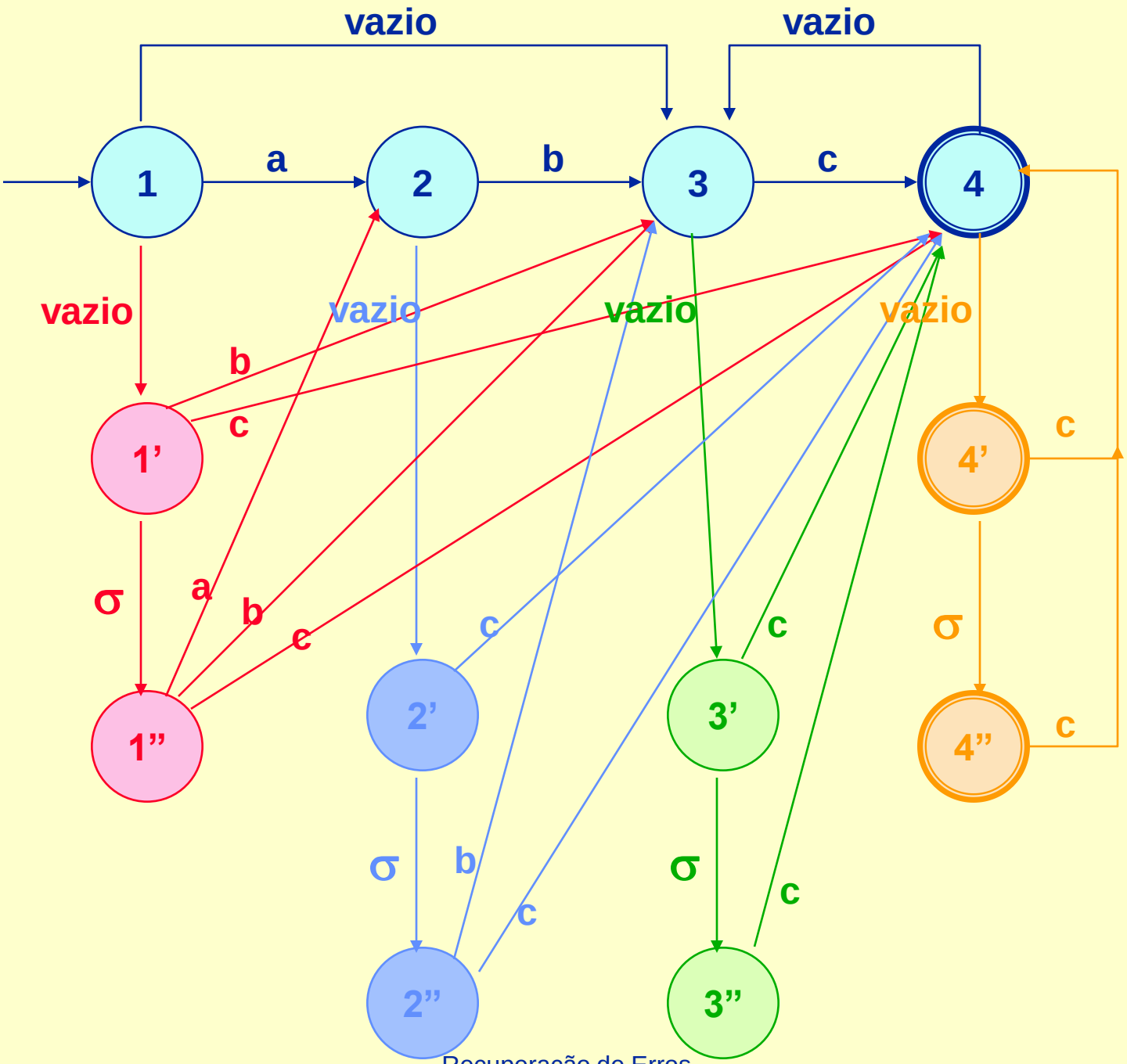
- **Faltam ao método apresentado procedimentos para o tratamento de:**
  - **transições em vazio**
  - **consumo múltiplo de um mesmo átomo a partir de um estado**
- **As demais regras apresentadas podem ser utilizadas de forma idêntica**
- **Os não-determinismos não afetam o método**
- **Isto acarreta duas alterações:**
  - **para obter o conjunto de sucessores de um estado i, colecionar os estados-destino de todas as cadeias de transições em vazio, seguidas de consumo de átomo**
  - **para múltiplas transições que, a partir de um estado i, resultem no consumo de um mesmo átomo, unir os conjuntos de sucessores correspondentes a cada uma destas transições.**

Exemplo



Levantamento das triplas

i=	triplas	átomos
1	(1 2 3)	a, b
1	(1 4 4)	c, c
2	(2 3 4)	b, c
3	(3 4 4)	c, c
4	(4 4 4)	c, c


vazio
vazio
1
a
2
b
3
c
4
vazio
vazio
vazio
vazio
b
c
1'
4'
c
σ
a
b
c
c
c
σ
1''
2'
3'
4''
c
σ
b
σ
c
c
2''
3''

Recuperação de erros múltiplos em autômatos finitos

- **Construído o recuperador de erros simples, podem ser acrescentadas extensões para prever erros múltiplos**
- **A recuperação absoluta de erros múltiplos é onerosa**
  - **não resolve o problema geral: contempla apenas um número finito de níveis de erros simultâneos**
  - **adiciona ao autômato um número de estados igual ao do autômato inicial para cada nível adicional de erros múltiplos**
  - **a cada novo nível, tendem a aumentar as ocorrências de não-determinismos, o que leva à redução da precisão do método**
- **O método conhecido como *"panic mode"* é uma solução aproximada bastante eficaz:**
  - **Trata-se de um método agressivo, cirúrgico**
  - **descarta do texto-fonte uma parte que seja suficiente para ressincronizar o autômato**
  - **É muito usado em compiladores comercialmente distribuídos**

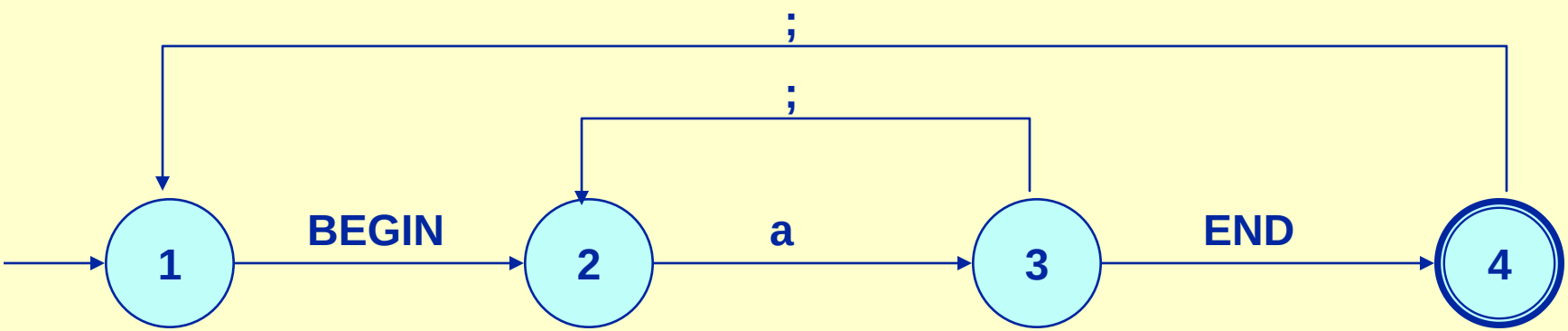
O método do *"panic mode"* básico

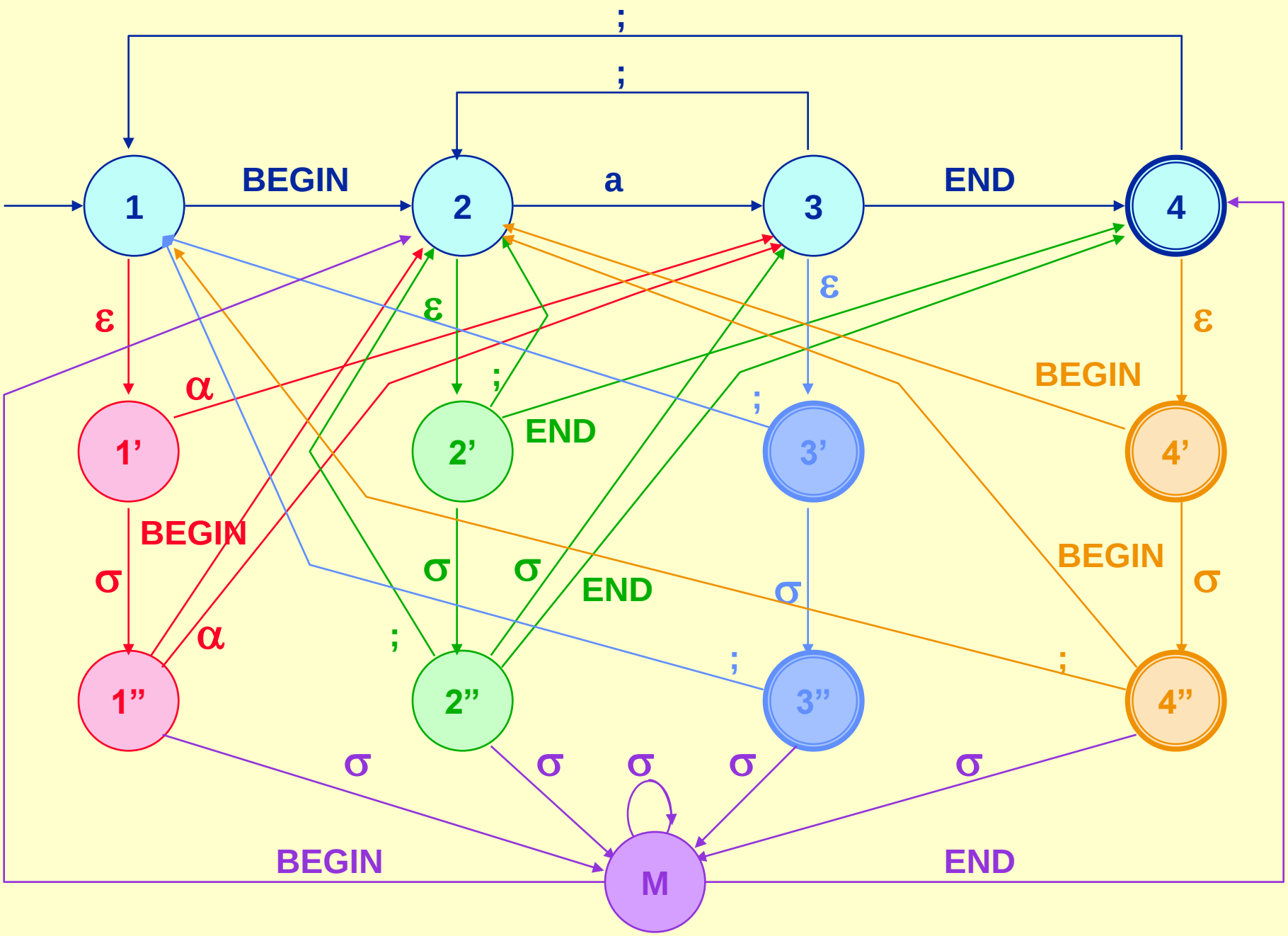
- **Utilizar o "panic mode" como complemento de outros métodos de recuperação de erros:**
- **Construir inicialmente o conjunto S de átomos de sincronização, com base em estatísticas**
- **Criar um novo estado M, onde átomos espúrios σ são eliminados: $(M, \sigma) \rightarrow M$**
- **O acesso a M se dá a partir de cada um dos estados i_2 criados na extensão de recuperação de erros simples: para cada i_2 cria-se $(i_2, \sigma) \rightarrow M$ consumindo átomos σ não pertencentes a S**
- **Para cada átomo de sincronização s em S, criar uma transição $(M, s) \rightarrow i_S$ onde i_S é o estado do autômato original, normalmente atingido após o consumo de s, para cada i_2 da extensão**

Exemplo

- **Linguagem:**

PROG = (“BEGIN” (“A” \ “;”) “END” \ “;”)



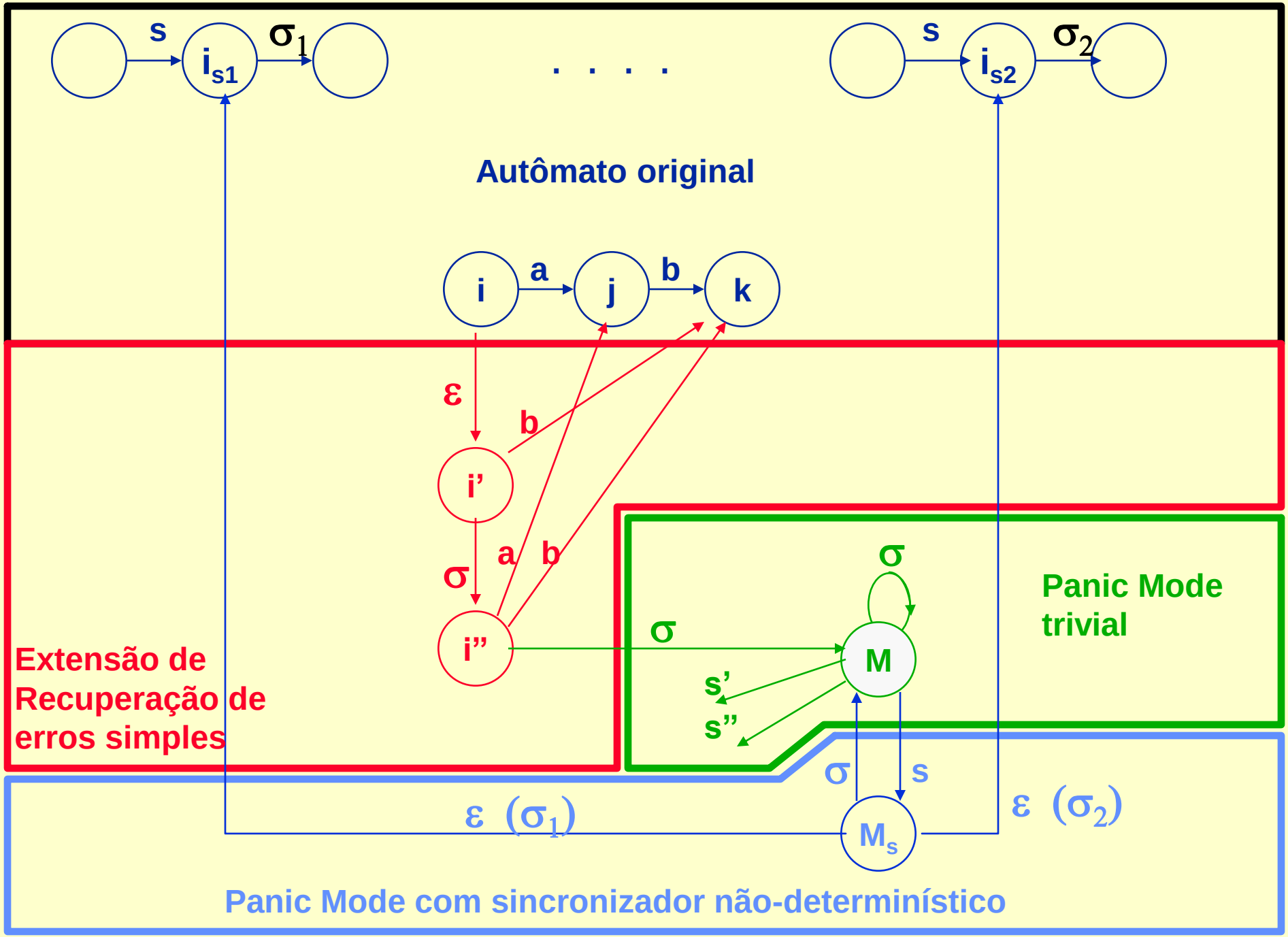

;
;
1
BEGIN
2
a
3
END
4
ε
ε
ε
ε
α
;
;
BEGIN
1'
2'
END
3'
4'
BEGIN
σ
σ
σ
END
σ
BEGIN
σ
α
;
;
;
1''
2''
3''
4''
σ
σ
σ
σ
σ
BEGIN
M
END

Observações

- **Ao autômato contendo extensões para a recuperação de erros simples, foram acrescentadas as transições responsáveis pela recuperação de erros múltiplos, escolhido o conjunto de átomos de sincronização S= { BEGIN , END }**
- **Notar a presença de redundâncias e não-determinismos no autômato obtido**
- **Notar ainda que o símbolo “;” , que poderia ser utilizado como sincronizador, provocaria um não-determinismo adicional na extensão de recuperação, já que no autômato original há duas transições que consomem tal símbolo, com estados-destino diferentes**

Comentários

- **Embora mencionado como método complementar de recuperação de erros, o “panic mode” é atraente e prático, podendo ser usado diretamente no autômato original, mas com perda de precisão**
- **Muitas vezes, átomos que poderiam ser usados como sincronizadores são encontrados em mais de uma transição com diferentes destinos no autômato original, prejudicando sua utilização pelos não-determinismos adicionais que geram**
- **Para tais casos, convém alterar a forma de construção da extensão, evitando o problema através de “look-ahead”, consultando-se o símbolo seguinte para escolher a transição a ser executada.**


s
i_{s1}
σ_1
. . . .
s
i_{s2}
σ_2
Autômato original
i
a
j
b
k
ε
b
i'
a
b
σ
i"
σ
σ
M
s'
s"
Panic Mode
trivial
Extensão de
Recuperação de
erros simples
σ
s
ε (σ1)
ε (σ2)
Ms
Panic Mode com sincronizador não-determinístico

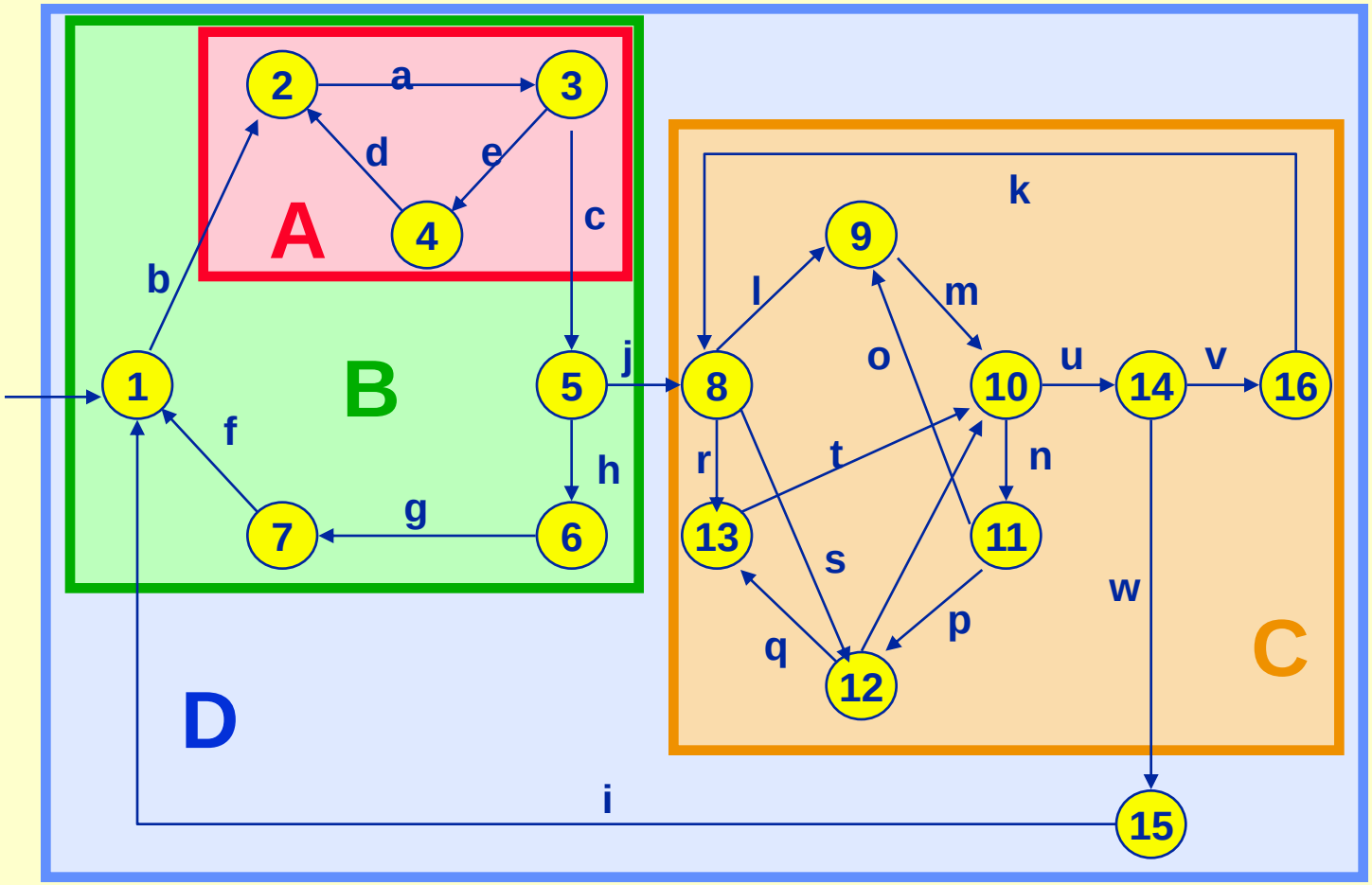
Um “panic-mode” melhorado (1)

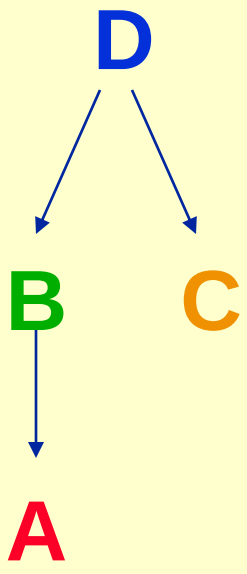
- **Com base no que foi estudado, e lembrando as características locais que as linguagens exibem, pode-se tornar conveniente aplicar o método básico do panic-mode de modo hierárquico, e não de forma global**
- **Esta hierarquização deve ser baseada na diferenciação sintática que se pode identificar entre as diversas partes do reconhecedor**
- **Para tanto, analisa-se inicialmente o autômato, identificando *grupos coesos*, conjuntos de estados que tomam parte do reconhecimento das estruturas sintáticas mais importantes da linguagem**
- **Em geral, os grupos coesos são formados por estados componentes dos loops que implementam construções repetitivas da linguagem**
- **Tais loops podem fazer parte de loops externos mais amplos, ou conter loops internos menores**

Um “panic-mode” melhorado (2)

- **Cada grupo coeso sugere um tratamento específico para erros detectados nos seus estados**
- **Pode-se, partindo do grupo coeso mais interno a que pertence o estado em que o erro se manifestou, procurar na cadeia de entrada um átomo sincronizador associado**
- **Caso o átomo corrente não atenda a este requisito, verifica-se se se trata de um átomo sincronizador do grupo coeso imediatamente mais externo, e assim por diante**
- **Fracassando-se em todos os níveis, descarta-se o átomo corrente, e repete-se o procedimento para o próximo átomo de entrada**
- **Aplica-se este procedimento a todos os estados do autômato que não exibam transições explícitas com todos os átomos do alfabeto de entrada, podendo-se usar ou não em conjunto outros métodos de recuperação**

Exemplo - grupos coesos





Hierarquia dos grupos coesos

Grupos coesos no autômato do exemplo:
A= {2,3,4} B={1,7,6,5,2,3,4} C={8,9,10,11,12,13,14,16}
D={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}

Exemplo (continuação)

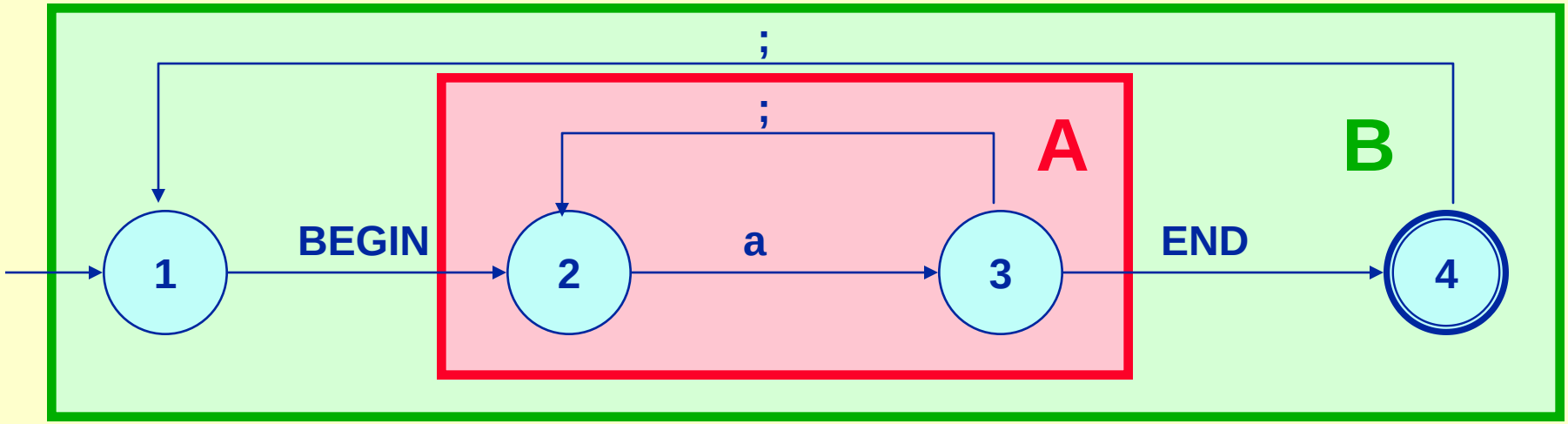
Grupos coesos	Conjunto de átomos de sincronização
A= { 2, 3, 4}	{ a }
B={ 1, 7, 6, 5, 2, 3, 4}	{ b, c, j }
C={ 8, 9, 10, 11, 12, 13, 14, 16}	{ u, w, l, r, s }
D={ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16}	{ w, i }

- Notar a escolha dos conjuntos de átomos de sincronização é relativamente livre, dando-se preferência aos átomos que iniciam, terminam ou realimentam as construções repetitivas
- Não são convenientes como sincronizadores os átomos que participam simultaneamente de mais de um grupo coeso.

- **Identificar situações de não-determinismo devidas à possível ressincronização em estados diversos causadas pelo mesmo átomo de ressincronização.**
- **Isto ocorre quando, entre grupos coesos do mesmo ramo da árvore de hierarquia, a intersecçào dos conjuntos de átomos sincronizadores não for vazia**
- **Para tais casos, criar estados intermediários para a implementação da parte da extensão responsável pela ressincronização do "panic-mode"não-determinístico**
- **Para cada grupo coeso criar um estado responsável pela ressincronização normal de "panic-mode"**
- **A partir de cada estado do autômato original, que não esgote as possibilidades de transições com consumo de átomos, criar uma transição partindo deste estado, com destino ao estado de "panic-mode" associado ao mais interno dos grupos coesos que contenha o estado. Esta transição deve consumir todos os átomos restantes.**

- **A partir do estado de ressincronização do grupo coeso em estudo, criar transições que utilizem o átomo corrente, consumindo-o caso pertença ao conjunto de átomos de ressincronização associado, e transitando para o correspondente estado do autômato principal**
- **Emitir, a partir deste estado, uma transição de "look-ahead" para cada átomo pertencente aos demais conjuntos dos átomos sincronizadores associados aos gurpos coesos de hierarquia mais alta que pertençam ao mesmo ramo na árvore de hierarquia. O destino destas transições deverá ser o estado-destino de consumo de átomo, associado aos átomos em questão, no autômato principal**
- **Emitir consumindo os demais átomos, nova transição com destino ao estado de ressincronização associado ao grupo coeso em que se manifestou o erro**
- **Repetir o procedimento para todos os estados do autômato**

Exemplo (1)



Grupos coesos:

A = { 2, 3 }

B = { 1, 2, 3, 4 }

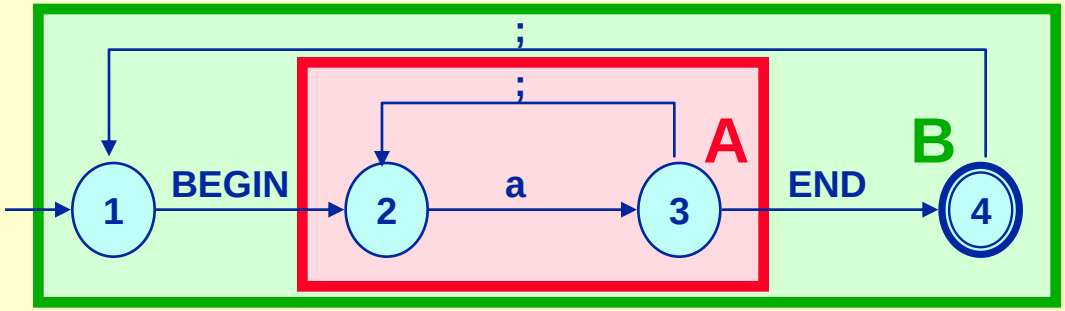
Sincronizadores:

S_A = { ; }

S_B = { BEGIN, END, ; }

Hierarquia:

Exemplo (2)

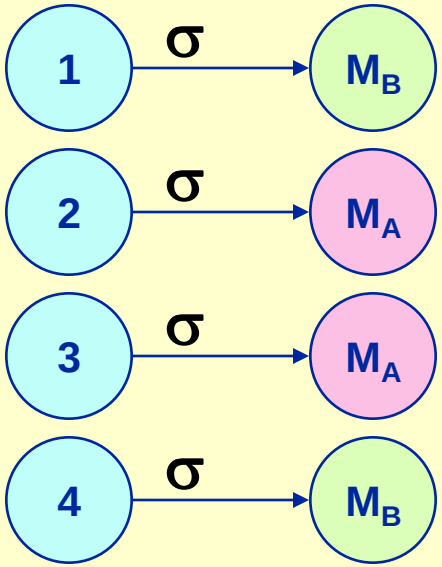


- **Não serão feitas recuperações absolutas neste exemplo**
- **Não-determinismos: Como não é vazia a intersecção entre os conjuntos S_A e S_B, o símbolo ";" deverá dar origem a um estado $M_;$**
- **Estados de ressincronização: M_A associado ao grupo coeso A, e M_B associado ao grupo coeso B**
- **Estados que não esgotam o alfabeto: 1, 2, 3, 4**

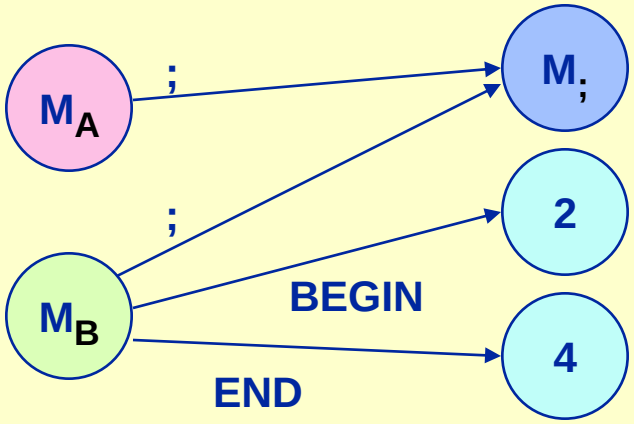
Exemplo (3)

- **Conexão aos estados de ressincronização:**

Estado	Grupo coeso mais local	Estado-destino
1	B	M_B
2	A	M_A
3	A	M_A
4	B	M_B

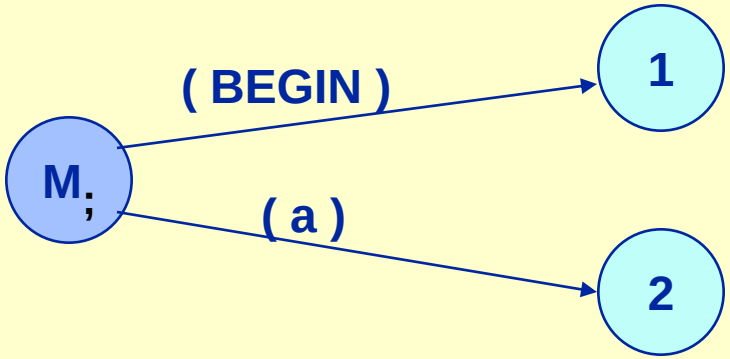


- **Uso dos átomos de sincronização:**

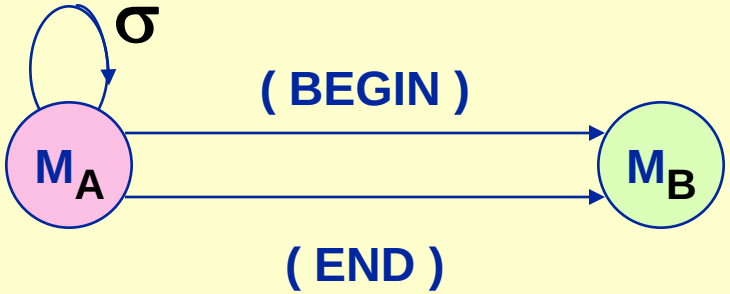


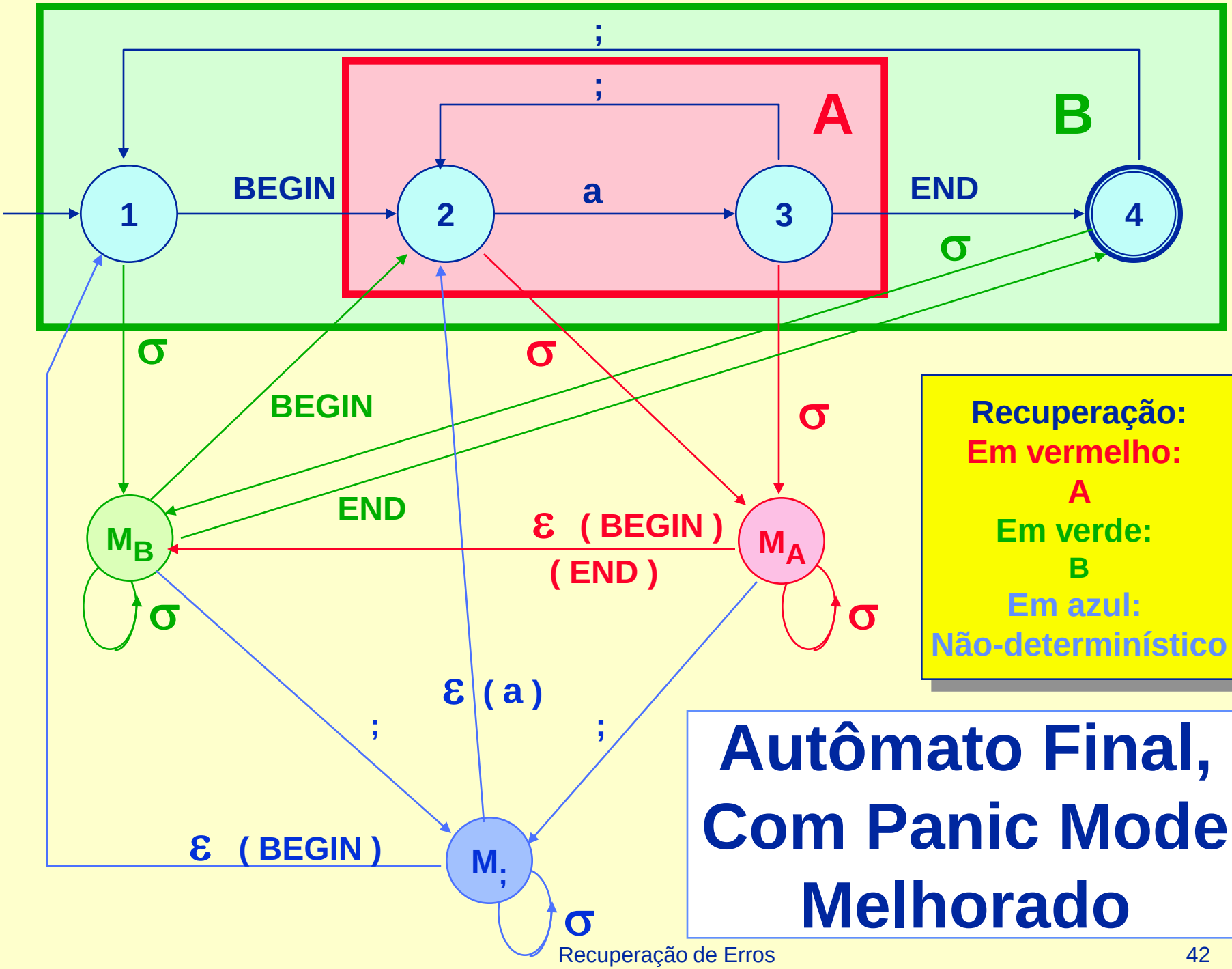
Exemplo (4)

- look-ahead para recuperação não-determinística



- “look-ahead” para mudança de nível entre grupos coesos





Recuperação:
Em vermelho:
A
Em verde:
B
Em azul:
Não-determinístico

Autômato Final, Com Panic Mode Melhorado

2. RECUPERAÇÃO DE ERROS EM AUTÔMATOS DE PILHA ESTRUTURADOS

Recuperação de Erros em Autômatos de Pilha Estruturados - caso determinístico

- **Aplica-se a técnica desenvolvida para autômatos finitos para tratar as transições internas contidas em todas as sub-máquinas do autômato. É preciso então acrescentar mecanismos para a cobertura das transições entre sub-máquinas**
- **A principal diferença reside na forma de determinação dos conjuntos de sucessores para um dado estado, pois o algoritmo deverá incluir os casos de chamada e de retorno de sub-máquinas, não considerados no caso de autômatos finitos por não existirem**

Determinação dos Primeiros Sucessores (1)

Determinação dos primeiros sucessores q_{xy} do estado q_{mi}

a) Sucessores relativos a transições internas:

Para seqüências do tipo

$$\gamma\ q_{mi}\ \alpha \rightarrow \gamma\ q_{x1}\ \alpha$$

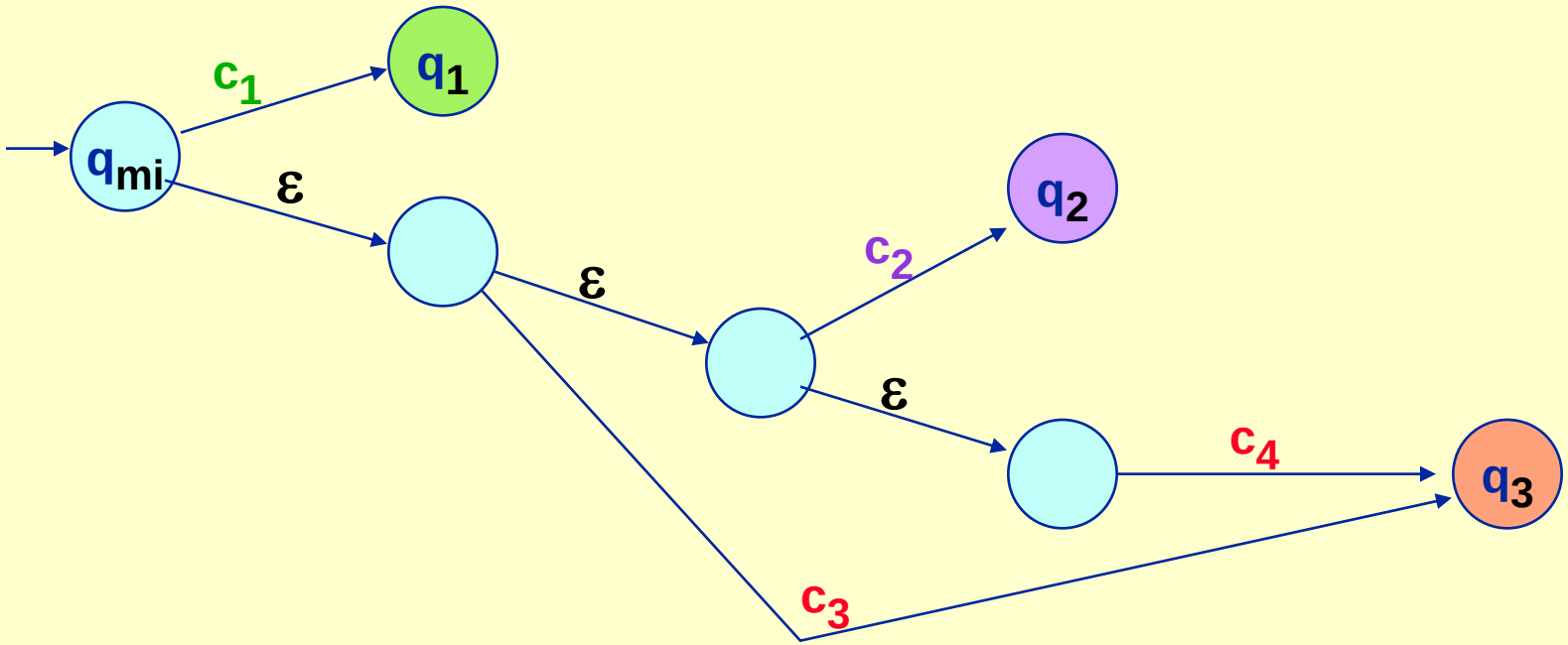
$$\ldots$$

$$\gamma\ q_{xn-1}\ \alpha \rightarrow \gamma\ q_{xn}\ \alpha$$

$$\gamma\ q_{xn}\ c\ \alpha \rightarrow \gamma\ q_{m\,j}\ \alpha$$

Incluir no conjunto de primeiros sucessores de q_{mi} os estados q_{mj} atingíveis após o consumo de átomos c

Exemplos: Primeiros Sucessores



ATOMOS (q_{mi}, q_1) = { c_1 }
ATOMOS (q_{mi}, q_2) = { c_2 }
ATOMOS (q_{mi}, q_3) = { c_3, c_4 }
PRIMEIRO (q_{mi}) = { c_1 } U { c_2 } U { c_3, c_4 } = { c_1, c_2, c_3, c_4 }

Determinação dos primeiros sucessores (2)

b) Sucessores relativos a retornos de sub-máquinas:

Seja neste caso q_{mi} um estado final.

Incorporar estados que sejam primeiros sucessores do estado q_{rs} para onde o retorno se efetua.

Para cada transição da forma

$$\gamma\ z_{rs}\ \ q_{mi}\ \alpha\ \rightarrow\ \gamma\ \ q_{rs}\ \alpha$$

Incluir todos os primeiros sucessores de q_{rs}

c) Sucessores relativos a chamadas de sub-máquinas:

Para cada transição da forma

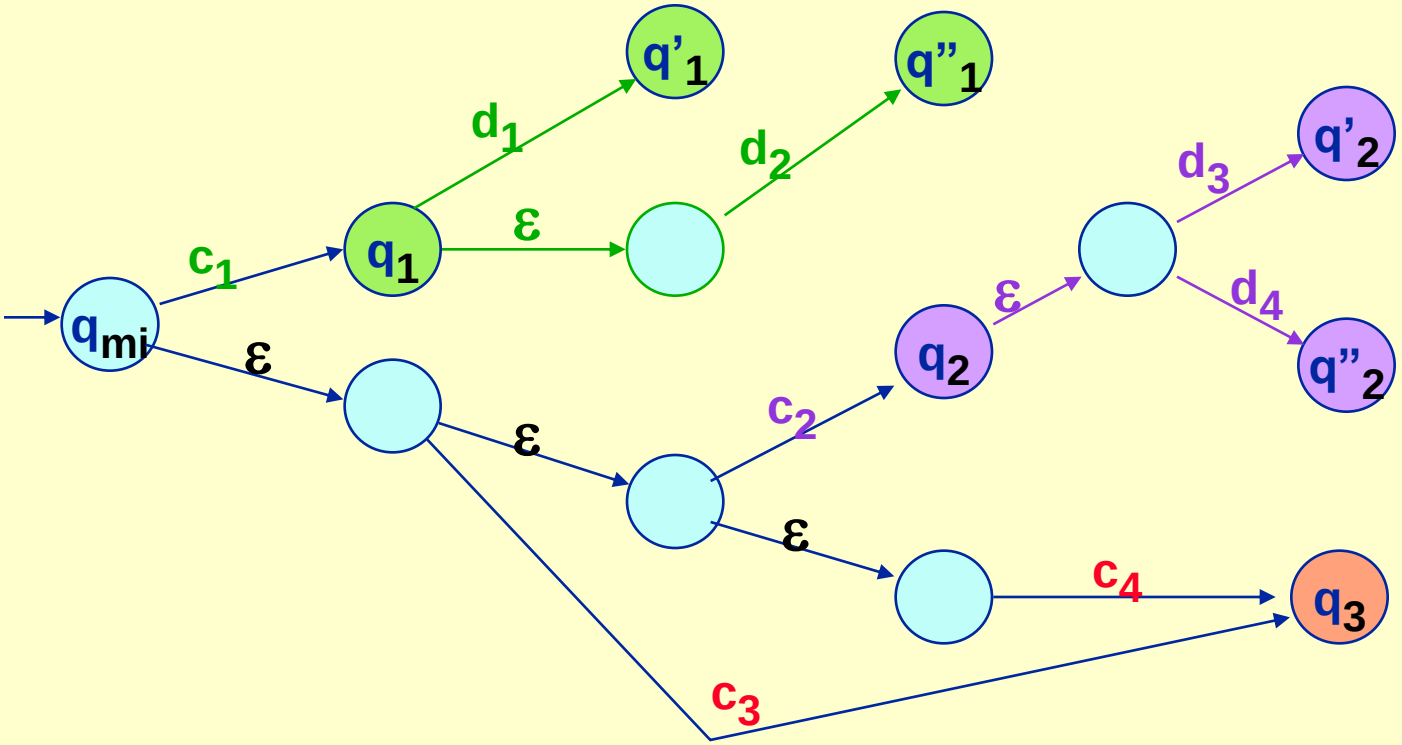
$$\gamma\ \ q_{mi}\ \alpha \rightarrow \gamma\ z_{mj}\ q_{r0}\ \alpha$$

Incluir todos os primeiros sucessores de q_{r0}

Algumas funções auxiliares

- **ATOMOS (q_{mi}, q_{xy}) - conjunto de átomos que o autômato consome em primeiro lugar partindo do estado q_{mi}, ao atingir o estado q_{xy}**
- **PRIMEIRO (q_{mi}) - conjunto-união de todos os conjuntos ATOMOS (q_{mi}, q_{xy})**
- **SEGUNDO (q_{mi}) - conjunto-união de todos os conjuntos PRIMEIRO (q_{xy})**
- **SEGUINTE (q_{mi}, c) - conjunto dos elementos de SEGUNDO (q_{mi}) atingidos após o consumo de c a partir do estado q_{mi}**

Exemplos: Segundos Sucessores e Seguintes



PRIMEIRO(q1) = {d1, d2}
PRIMEIRO (q2) ={d3, d4}
PRIMEIRO(q3) = { }
SEGUNDO(qmi) = {d1, d2} U {d3, d4} U { } = {d1, d2, d3, d4}
SEGUINTE(qmi, c2) = {d3, d4}
SEGUINTE(qmi, c3) = { }

Hipóteses:

- **q_{mi} - cada ponto de potencial manifestação do erro**

- **q_{mj} - estado-destino de alguma transição da sub-máquina a_m com origem em q_{mi} ou estado de retorno de alguma chamada de sub-máquina originada no estado q_{mi}**

$$\gamma\ q_{mi}\ \alpha \rightarrow \gamma\ q_{mj}\ \alpha$$

$$\gamma\ q_{mi}\ c\alpha \rightarrow \gamma\ q_{mj}\ \alpha$$

$$\gamma\ q_{mi}\ \alpha \rightarrow \gamma\ z_{mj}\ q_{n0}\ \alpha$$

- **q_{mk} - estado-destino de alguma transição originada em q_{mj} ou estado de retorno de alguma transição de chamada de sub-máquina, com origem em q_{mj}**

$$\gamma\ q_{mj}\ \alpha \rightarrow \gamma\ q_{mk}\ \alpha$$

$$\gamma\ q_{mj}\ b\alpha \rightarrow \gamma\ q_{mk}\ \alpha$$

$$\gamma\ q_{mj}\ \alpha \rightarrow \gamma\ z_{mk}\ q_{n0}\ \alpha$$

- **Serão criados estados de recuperação q_{mi}' e q_{mi}"**

Recuperação interna à sub-máquina

- **Para cada q_{mi} criam-se as seguintes transições:**
  - **para ligar o autômato à extensão, e isolá-lo da mesma:**

$$\gamma\ q_{mi}\ \alpha \rightarrow \gamma\ q_{mi}'\ \alpha$$

  - **para descartar átomos σ espúrios (erros de inserção ou de substituição), [sendo σ não-pertencente a ATOMOS (q_{mi}, q_{mj})] :**

$$\gamma\ q_{mi}'\ \sigma\alpha \rightarrow \gamma\ q_{mi}''\ \alpha$$

  - **recuperar casos de inserção [c pertence a ATOMOS (q_{mi}, q_{mj})] :**

$$\gamma\ q_{mi}''\ c\alpha \rightarrow \gamma\ q_{mj}\ \alpha$$

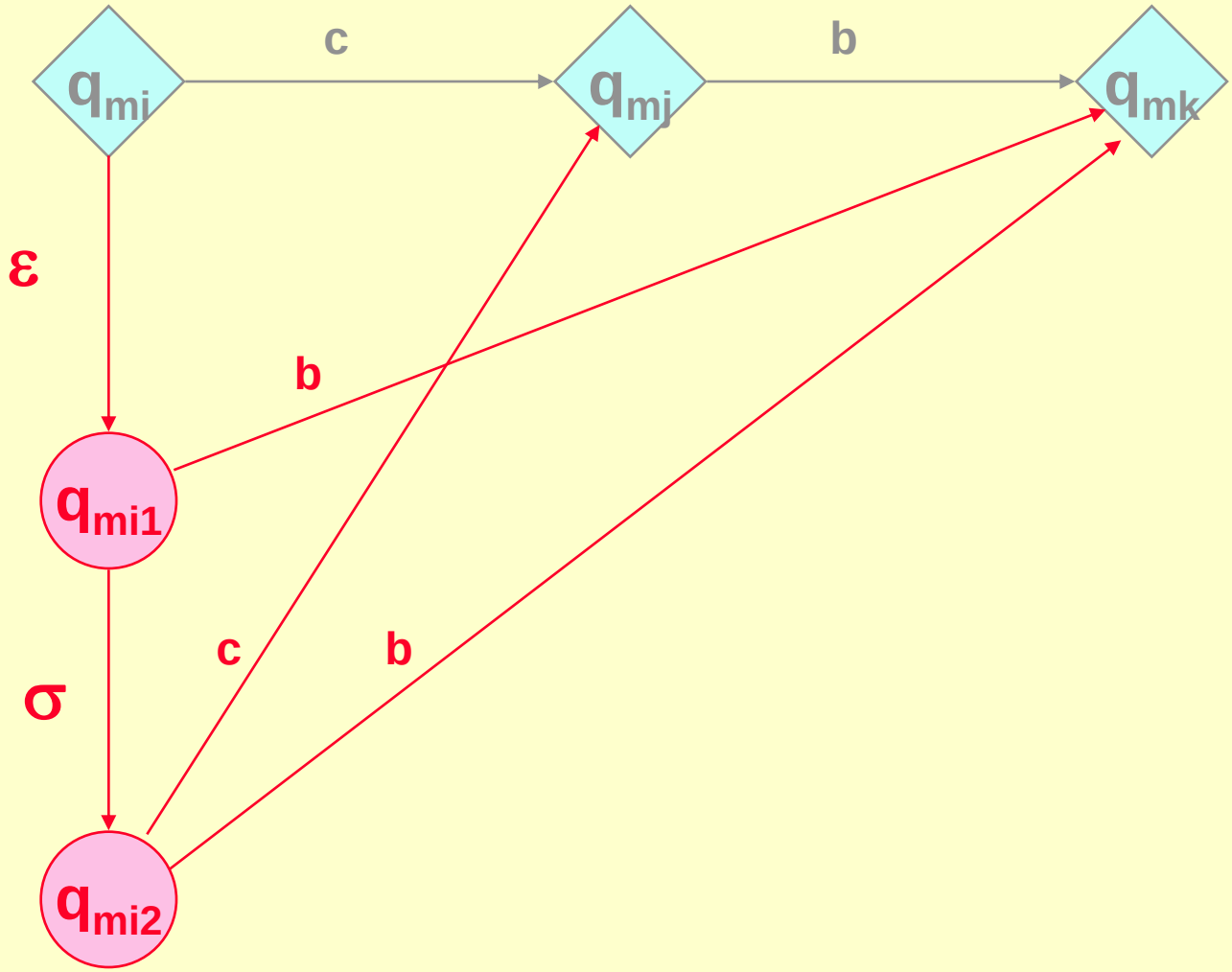
  - **erros de eliminação e substituição [b pertence a ATOMOS (q_{mi}, q_{mj})]:**

$$\gamma\ q_{mi}'\ b\alpha \rightarrow \gamma\ q_{mk}\ \alpha$$

$$\gamma\ q_{mi}''\ b\alpha \rightarrow \gamma\ q_{mk}\ \alpha$$

- **O procedimento acima é idêntico ao que foi apresentado para autômatos finitos**

Exemplo - Recuperação interna



Recuperação em estados finais de sub-máquina

- **Se q_{mi} é estado final, existirá uma transição do tipo:**

$$\gamma\, z_{rs}\ \ q_{mi}\ \alpha \ \rightarrow\ \gamma\ \ q_{rs}\ \alpha$$

  - substituem-se tais transições pelas seguintes:

$$\gamma\, z_{rs}\ \ q_{mi}\ c\alpha \ \rightarrow\ \gamma\ \ q_{rs}\ c\alpha$$

    para cada c do conjunto PRIMEIRO (q_{rs})
  - Incluir, correspondentemente, transições do tipo

$$\gamma\, z_{rs}\ \ q_{mi}''\ c\alpha \ \rightarrow\ \gamma\ \ q_{rs}\ c\alpha$$

    para recuperar erros de inserção em q_{mi}
  - Para recuperar erros de omissão e de substituição, incuem-se, para cada átomo d em SEGUNDO (q_{rs})

$$\gamma\, z_{rs}\ \ q_{mi}'\ d\alpha \ \rightarrow\ \gamma\ \ q_{rs}\ d\alpha$$

$$\gamma\, z_{rs}\ \ q_{mi}''\ d\alpha \ \rightarrow\ \gamma\ \ q_{rs}\ d\alpha$$

- **Se q_{mj} é estado final:**

$$\gamma\, z_{rs}\ q_{mj}\ \alpha \ \rightarrow\ \gamma\ q_{rs}\ \alpha$$

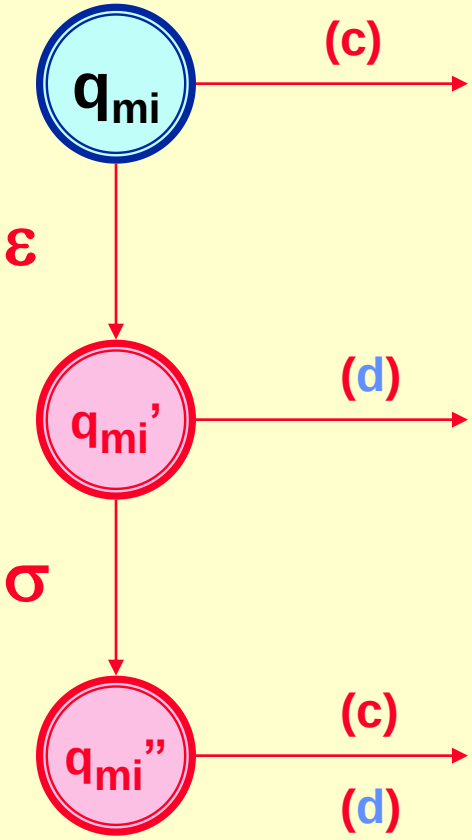
  - substituem-se tais transições pelas seguintes:

$$\gamma\, z_{rs}\ \ q_{mj}'\ c\alpha \ \rightarrow\ \gamma\ \ q_{rs}\ c\alpha$$

$$\gamma\, z_{rs}\ \ q_{mj}''\ c\alpha \ \rightarrow\ \gamma\ \ q_{rs}\ c\alpha$$

    para cada c do conjunto PRIMEIRO (q_{rs})

Exemplo - Recuperação em estados finais



- Retornos condicionais a partir de q_{mi} com look-ahead dos átomos **c** pertencentes a PRIMEIRO (q_{rs})
- Retornos condicionais inseridos a partir de q_{mi}" com look-ahead dos átomos **c** pertencentes a PRIMEIRO(q_{rs})
- Retornos condicionais a partir de q_{mi}' e q_{mi}" com look-ahead dos átomos **d** pertencentes a SEGUNDO(q_{rs})

Caso em que q_{mi} é estado final

Recuperação em chamadas de sub-máquinas (1)

- **Para as chamadas de sub-máquinas, da forma:**

$$\gamma\ q_{mi}\ \alpha \rightarrow \gamma\ z_{mj}\ q_{n0}\ \alpha$$

  - substituem-se tais transições pelas seguintes:

$$\gamma\ q_{mi}\ c\alpha \rightarrow \gamma\ z_{mj}\ q_{n0}\ c\alpha$$

    para cada c do conjunto PRIMEIRO (q_{n0})

  - Para recuperar erros de inserção, incluir:

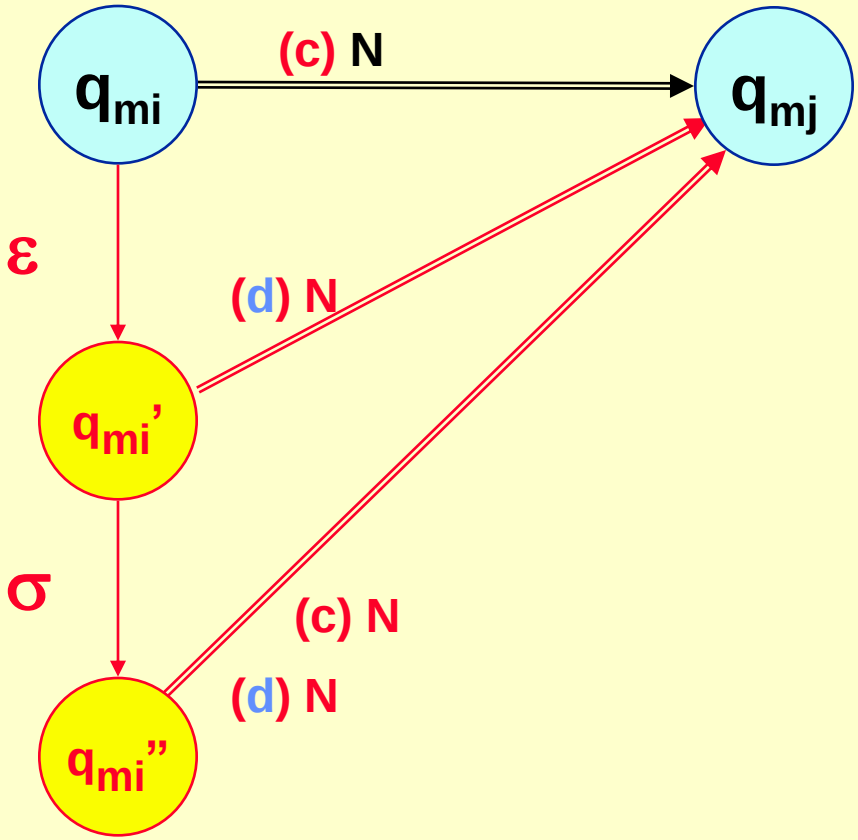
$$\gamma\ q_{mi}''\ c\alpha \rightarrow \gamma\ z_{mj}\ q_{n0}\ c\alpha$$

  - Para recuperar erros de omissão e de substituição, incluem-se, para cada átomo d em SEGUNDO (q_{rs}):

$$\gamma\ q_{mi}'\ d\alpha \rightarrow \gamma\ z_{mj}\ q_{n0}\ d\alpha$$

$$\gamma\ q_{mi}''\ d\alpha \rightarrow \gamma\ z_{mj}\ q_{n0}\ d\alpha$$

Exemplo - Recuperação em chamadas de sub-máquina



- Chamadas condicionais da sub-máquina N a partir de q_{mi}" com look-ahead dos elementos **c** do conjunto PRIMEIRO(q_{n0})
- Chamadas condicionais da sub-máquina N a partir de q_{mi}' e q_{mi}" com look-ahead dos elementos **d** do conjunto SEGUNDO(q_{n0})

Recuperação em chamadas de sub-máquinas (2)

- **Caso haja um estado q_{mj} que seja origem de chamadas de sub-máquina $a_{n'}$, tem-se**

$$\gamma\ q_{mj}\ \alpha\ \rightarrow\ \gamma\ z_{mk}\ q_{n'0}\ \alpha$$

- **Havendo no mínimo uma transição com consumo de átomo partindo de q_{mi} e atingindo este estado**

$$\gamma\ q_{mi}\ b\alpha\ \rightarrow\ \gamma\ q_{mj}\ \alpha$$

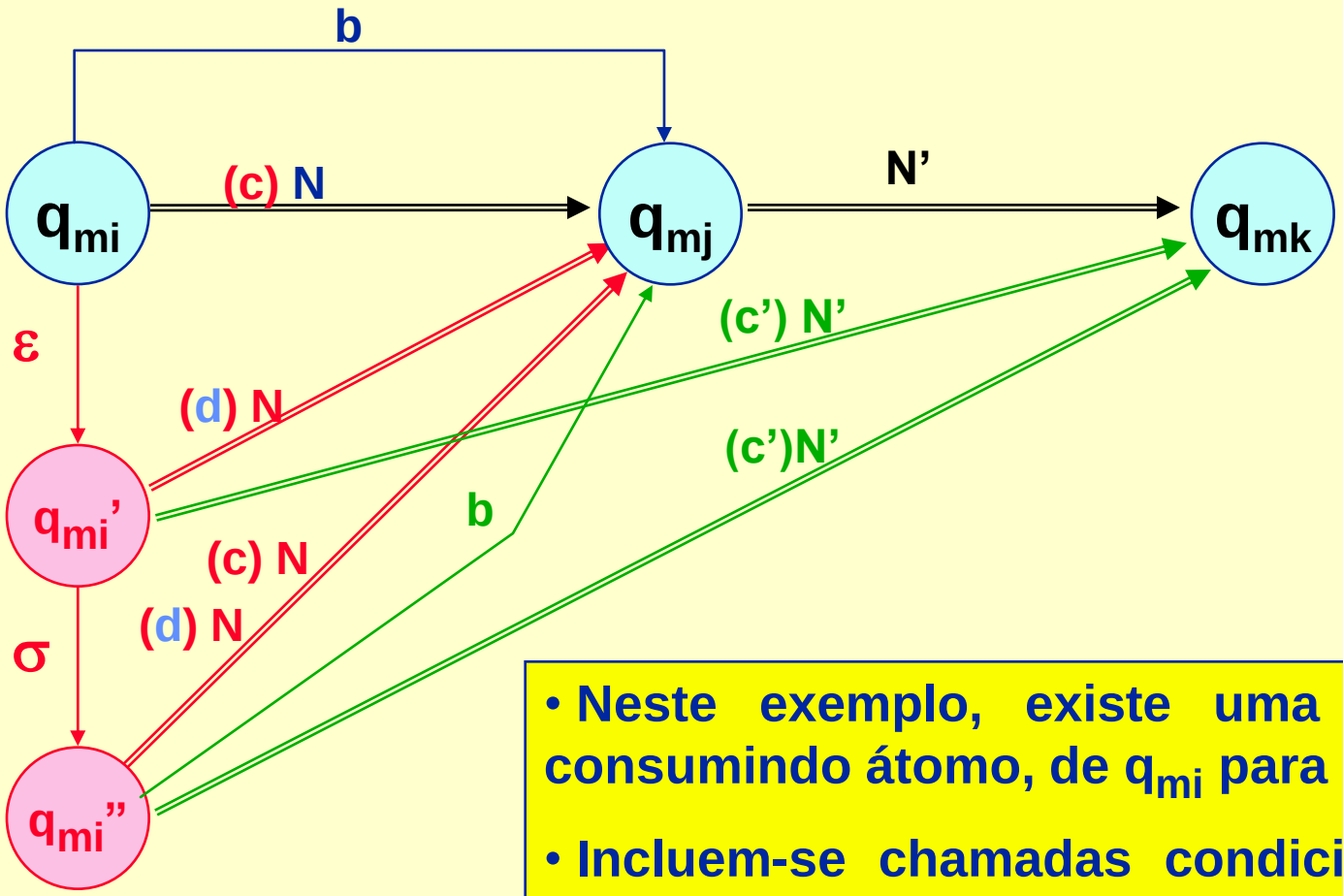
Incluir, para cada c' em PRIMEIRO($q_{n'0}$), onde $q_{n'0}$ é o estado inicial da sub-máquina $a_{n'}$, as transições

$$\gamma\ q_{mj'}\ c'\alpha\ \rightarrow\ \gamma\ z_{mk}\ q_{n'0}\ c'\alpha$$

$$\gamma\ q_{mj''}\ c'\alpha\ \rightarrow\ \gamma\ z_{mk}\ q_{n'0}\ c'\alpha$$

- **Se não houver transições consumindo átomos entre q_{mi} e q_{mj}, não incluir qualquer transição de recuperação relativa à chamada de $a_{n'}$ em q_{mj}.**

Exemplo



- Neste exemplo, existe uma transição, consumindo átomo, de q_{mi} para q_{mj}
- Incluem-se chamadas condicionais de $a_{N'}$ a partir dos dois estados $q_{mi'}$ e $q_{mi''}$, com look-ahead dos átomos c' do conjunto PRIMEIRO($q_{n'0}$)

Panic Mode

- **Cria-se inicialmente um estado de Panic Mode para cada sub-máquina, no qual átomos espúrios σ são eliminados**

$$\gamma\ q_{mM}\ \sigma\alpha \rightarrow \gamma\ q_{mM}\ \alpha$$

Os áatomos eliminados não fazem parte do conjunto S de átomos de sincronização adotado para a gramática

- **O acesso ao estado q_{mM} se dá a partir dos estados q_{mi}" criados para a recuperação de erros simples, consumido nesta transição átomos σ não pertencentes a S:**

$$\gamma\ q_{mi}"\ \sigma\alpha \rightarrow \gamma\ q_{mM}\ \alpha$$

- **A partir de q_{mM} emergem adicionalmente, com destino a estados internos, transições em que são consumindo átomos s de S, tais que os símbolos s sejam consumido na própria sub-máquina:**

$$\gamma\ q_{mu}\ s\alpha \rightarrow \gamma\ q_{mv}\ \alpha$$

- **Para cada uma delas, a transição de recuperação assume a forma:**

$$\gamma\ q_{mM}\ s\alpha \rightarrow \gamma\ q_{mM}\ \alpha$$

- **Para cada s de S, que não seja consumido localmente, mas para o qual exista alguma transição do tipo**

$$\gamma\ q_{m'u}\ s\alpha \rightarrow \gamma\ q_{mM}\ \alpha$$

Criam-se transições de retorno com desempilhamento forçado:

$$\gamma\ r_{m'x}\ q_{mM}\ s\alpha \rightarrow \gamma\ q_{m'v}\ \alpha$$

Recuperação de erros sintáticos nos analisadores determinísticos clássicos

- **Para uma cadeia AXB onde A representa a parte da cadeia de entrada já consumida, X o átomo corrente que provocou a dessincronização, e B a parte da cadeia ainda não analisada, os métodos comumente empregados para a recuperação de erros sintáticos fundamentam-se principalmente em algum(ns) dos seguintes procedimentos:**
  - **remover o átomo X causador da manifestação do erro**
  - **inserir entre A e X uma seqüência C de átomos tal que se torne possível consumir CX sem que novo erro seja detectado**
  - **eliminar de $A=A_1A_2$ o sufixo A_2, de tal maneira que ao ser reanalisada a cadeia resultante A_1XB o átomo X seja consumido sem que um novo erro seja detectado**

3. RECUPERAÇÃO DE ERROS EM ANALISADORES DESCENDENTES

Recuperação em Analisadores Descendentes (1)

- **Em analisadores descendentes, a análise se processa através da montagem da árvore de reconhecimento a partir da raiz em direção às folhas**
- **Assim, analisadores descendentes são de uma certa forma preditivos, indicando a priori, através dos não-terminais ainda não desenvolvidos, o caminho para completar o reconhecimento a partir do conteúdo esperado do restante do texto-fonte**
- **Para elaborar um recuperador de erros neste caso, pode-se construir um conjunto S de símbolos da gramática que iniciam a definição de algum não-terminal pendente presente no contorno da árvore em construção**

Recuperação em Analisadores Descendentes (2)

- **Detectado o erro, descartam-se símbolos até que seja possível derivar, a partir de algum dos não-terminais pendentes, o texto-fonte restante**
- **Insere-se então, em lugar da seqüência eliminada, uma cadeia que seja conveniente para preencher a parte da árvore compreendida entre o não-terminal que permitiu prosseguir o reconhecimento e o último não-terminal corretamente desenvolvido**
- **Isto feito reinicia-se o reconhecimento, a partir da subcadeia inserida pelo método**

Recuperação de erros múltiplos em reconhecedores descendentes recursivos (1)

- **Reconhecedores desecendentes recursivos não constroem fisicamente a árvore de derivação, nem utilizam uma pilha sintática explícita.**
- **Em lugar disso, a pilha representa um conjunto de endereços de retorno, e faz parte do ambiente de execução da linguagem em que o autômato estiver sendo implementado**
- **Os estados do reconhecedor são realizados, neste caso, pelos conjuntos de registros de ativação das rotinas que implementam os não-terminais da gramática**

Recuperação de erros múltiplos em analisadores descendentes recursivos (2)

- **Considerando cada rotina (não-terminal), podem-se identificar conjuntos correspondentes de átomos sincronizadores locais, de modo análogo ao que foi estudado para o panic-mode dos autômatos finitos**
- **Globalmente, a recuperação se dá da seguinte forma:**
  - **Manifestado o erro, desvia-se para um ponto interno à rotina, onde será efetuada a recuperação local.**
  - **Se o átomo pertencer ao correspondente conjunto dos átomos de ressincronização local, desvia-se para o ponto associado da rotina, e prossegue-se normalmente o reconhecimento.**
  - **Se não, verifica-se se o átomo pertence a um outro conjunto recebido como parâmetro pela rotina corrente, a partir da rotina chamadora. Este deve ser o conjunto-união de todos os conjuntos de recuperação local das rotinas ativas no momento.**
  - **Caso o átomo pertença a este conjunto, retorna-se à rotina chamadora com a indicação do erro, se não descarta-se o átomo e analisa-se o átomo seguinte até que seja possível a ressincronização ou o retorno com indicação de erro.**

Exemplo

- **Para a seqüência de chamadas encadeadas das rotinas R_1, R_2, ... , R_n a partir da raiz R_0 tem-se:**

Rotina corrente	Sincronizadores locais	Conjunto global
programa principal	nenhum	vazio
raiz	S_{R_0}	vazio
R_1	S_{R_1}	S_{R_0}
R_2	S_{R_2}	$S_{R_0} \cup S_{R_1}$
. . .	. . .	. . .
R_n	S_{R_n}	$S_{R_0} \cup S_{R_1} \cup ... \cup S_{R_{n-1}}$

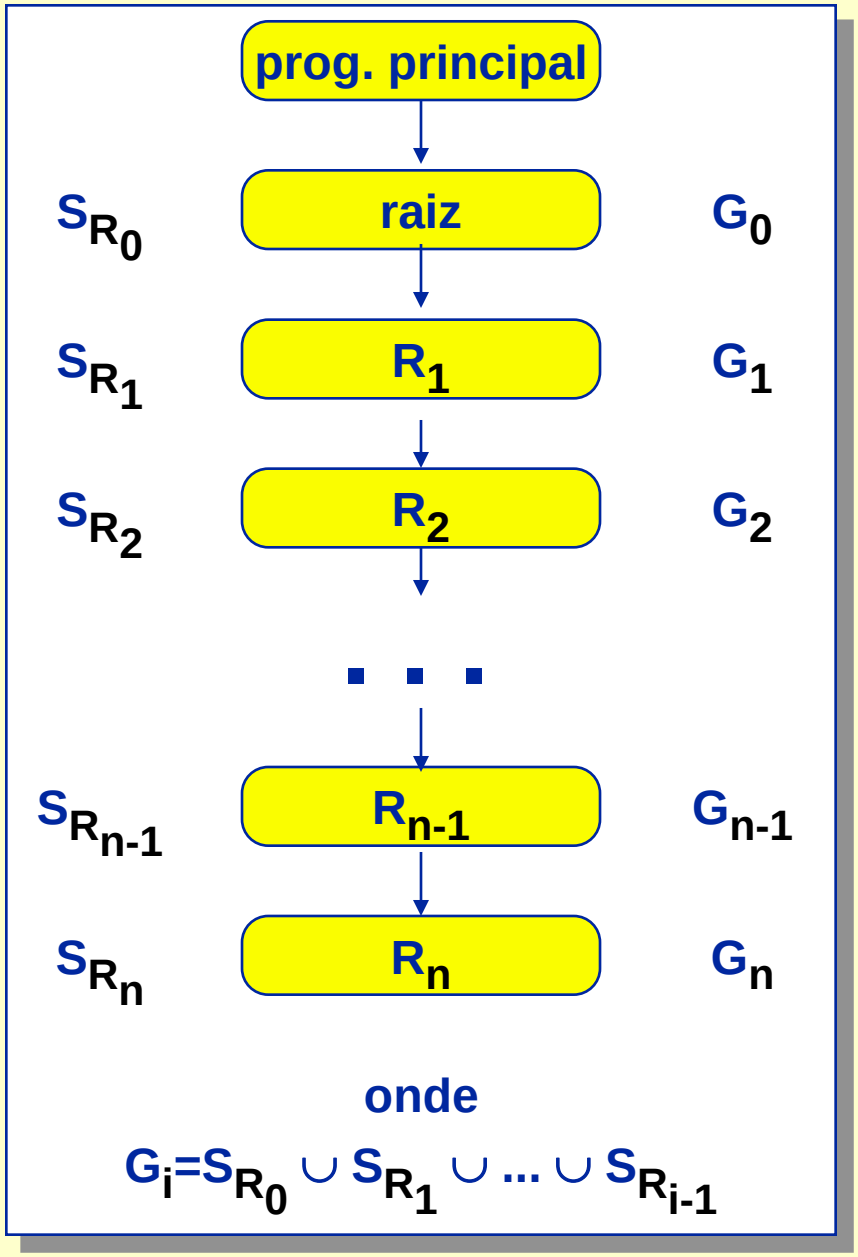
Exemplo (2)

Rotina R_i :

parâmetros: G_i, condição de erro
Conjunto de recuperação local: S_{R_i}
Conjunto de recuperação global: G_i

$X_{\text{átomo } j}$:

RECUPERAÇÃO:
átomo pertence ao conjunto S_{R_i} ?
Sim: Desvia para o rótulo correspondente $X_{\text{átomo } j}$
Não: átomo pertence a G_i ?
Sim: retorna, com condição de erro, para a rotina chamadora R_{i-1}
Não: descarta átomo e desvia para o rótulo RECUPERAÇÃO



Recuperação de Erros na Análise LL

- **Nas tabelas LL, células vazias correspondem a situações de erro, podendo ser usadas para ativar rotinas específicas de recuperação de erros.**
- **Durante o reconhecimento preditivo, a ocorrência de um erro é detectada quando:**
  - **Um terminal no topo da pilha não coincide com o símbolo em análise da cadeia de entrada, ou então**
  - **Há um não-terminal A no topo da pilha, α é o símbolo em análise da cadeia de entrada, e M[A,α] indica uma situação de erro (a célula correspondente da tabela de análise M está em branco).**

- **Pode-se alterar a tabela de análise para recuperar erros segundo dois processos diferentes:**
  - ***Panic Mode*: o analisador despreza símbolos da entrada até encontrar um átomo de sincronização.**
  - **Recuperação local (para erros simples): o analisador atua fazendo alterações sobre um símbolo apenas:**
    - » **desprezando o átomo de entrada em análise, ou**
    - » **substituindo-o por outro, ou**
    - » **inserindo um novo átomo, ou ainda,**
    - » **removendo um símbolo da pilha.**

Panic Mode

- **Baseia-se na idéia de ignorar átomos da cadeia de entrada até encontrar um que pertença a um conjunto de átomos de sincronização.**
- **A eficácia desse método de recuperação de erros depende da escolha cuidadosa desse conjunto de sincronização.**
- **O conjunto de sincronização deve ser escolhido de tal modo que o analisador se recupere rapidamente dos erros que com maior frequência ocorrem na prática.**

- **Quando bem implementado costuma ser um método muito eficaz para recuperação de erros.**
- **Considerando que a cadeia de entrada é finita, e que a cada intervenção do algoritmo de *panic mode* um dos seus átomos é descartado, pode-se garantir que, após um número finito de intervenções, certamente o procedimento de recuperação termina.**
- **Associa a cada procedimento recursivo um parâmetro adicional composto por um conjunto de marcas de sincronização.**

- **Os átomos que podem restaurar a sincronização são acrescentados a esse conjunto cada vez que ocorre uma ativação.**
- **Encontrado um erro, o procedimento de recuperação tem início e o analisador varre os átomos à frente na cadeia de entrada, descartando átomos, até que algum dos átomos pertencentes ao conjunto de sincronização seja encontrado na cadeia de entrada, encerrando a recuperação.**

- **Inclua todos os símbolos do FOLLOW(A) no conjunto de sincronização para o não-terminal A.**
- **Acrescentar ao conjunto de sincronização de uma construção de nível inferior os símbolos que iniciam construções de nível superior.**
- **Incluir os símbolos em FIRST(A) no conjunto de sincronização do não-terminal A (pode retornar a análise de acordo com A se um símbolo em FIRST(A) aparecer na entrada).**
- **Se um não-terminal gerar a cadeia vazia, pode adiar a detecção de erro, mas não faz com que o erro se perca.**
- **Se um terminal no topo da pilha não casar com o terminal da entrada, desempilha o terminal.**

Exemplo

Entrada:
id+id*id $

$E \rightarrow T\ E'\ \$$

$E' \rightarrow +\ T\ E' \mid \varepsilon$

$T \rightarrow F\ T'$

$T' \rightarrow *\ F\ T' \mid \varepsilon$

$F \rightarrow (E) \mid id$

FIRST(F) = FIRST(T) = FIRST(E) = {(, id}

FIRST(E') = {+, ε}

FIRST(T') = {*, ε}

FOLLOW(E) = FOLLOW(E') = {), $}

FOLLOW(T) = FOLLOW(T') = {+,), $}

FOLLOW(F) = {+, *,), $}.

	id	+	*	(	)	$
E	$E \rightarrow TE'\$$			$E \rightarrow TE'\$$	synch	synch
E'		$E' \rightarrow +TE'$			$E' \rightarrow \varepsilon$	$E' \rightarrow \varepsilon$
T	$T \rightarrow FT'$	synch		$T \rightarrow FT'$	synch	synch
T'		$T' \rightarrow \varepsilon$	$T' \rightarrow *FT'$		$T' \rightarrow \varepsilon$	$T' \rightarrow \varepsilon$
F	$F \rightarrow id$	synch	synch	$F \rightarrow (E)$	synch	synch

- Se o analisador sintático encontrar uma célula M[A, a] em branco, então o átomo a da cadeia de entrada é ignorado.
- Se a célula contém o indicador "synch", então o não-terminal presente no topo da pilha é desempilhado, na tentativa de retomar a análise.
- Se um *átomo* no topo da pilha não coincidir com o indicado na célula, então desempilha-se o *átomo do topo* da pilha, conforme mencionado anteriormente

4. RECUPERAÇÃO DE ERROS EM ANALISADORES ASCENDENTES

Recuperação de Erros em Analisadores Ascendentes

- **No caso dos analisadores ascendentes, a recuperação de erros deve ser feita nas imediações do seu ponto de detecção, visto não se dispor de informações mais globais acerca do texto-fonte, como ocorre com os analisadores descendentes que pela própria natureza são preditivos.**

- **O método seguinte, que exige uma operação de *look-ahead* de um átomo, aplica-se a analisadores nos quais seja possível realimentar o analisador léxico com símbolos determinados pelo mecanismo de análise sintática e de recuperação de erros.**

Recuperação Ascendente

- **Seja X um átomo que provocou a manifestação de erro.**
  A pilha sintática, com S_0 no seu topo, tem então a forma

$$S=S_nS_{n-1}...S_{j+1}S_jS_{j-1}....S_1S_0$$

  - **Ligar a variável ERRO, indicando recuperação em curso**
  - **Salvar o conteúdo da pilha**
  - **Iniciar j com 0 para apontar o topo da pilha. Para restaurar a pilha, obtém-se $S=S_nS_{n-1}...S_{j+1}S_j$ (j indica o número de elementos desempilhados).**
  - **Inserção de Terminal - Para todo terminal T aceito no estado S_j, inserir T antes de X, e tentar reconhecer a cadeia resultante. Se for possível, a recuperação foi bem sucedida. Se não, restaurar a pilha e continuar.**
  - **Inserção de não-terminal - Se S' for um estado tal que existe uma transição de S_j para S' no autômato LR, empilhar S' e tentar reconhecer a cadeia de entrada. Se possível, encontrou-se uma recuperação. Se não, restaurar a pilha e continuar.**
  - **Eliminação de estado - se j=n, executar o item seguinte, caso contrário, incrementar j e tentar novamente o reconhecimento. Se não for possível, voltar ao passo de inserção de terminal.**
  - **Pilha vazia - Aqui, j=n. Fazer j=0 e restaurar a pilha. Ler o próximo átomo, chamá-lo X e voltar ao passo de inserção de terminal.**

Exemplo: Recuperação Local Ascendente

(baseado em http://www.cs.princeton.edu/courses/archive/spring15/cos320/)

exp : NUM	**()**	**exps : exp**	**()**
| exp PLUS exp	**()**	**| exps ; exp**	**()**
| LPAR exp RPAR	**()**		

- Uma estratégia geral, válida tanto para análise *bottom-up* como para *top-down*, é a busca de um átomo de sincronização, ou seja, a estratégia de ***panic mode***

exp : NUM ()
| exp PLUS exp ()
| LPAR exp RPAR ()

exps : exp ()
| exps ; exp ()

- Uma estratégia geral, válida tanto para análise *bottom-up* como para *top-down*, é a busca de um átomo de sincronização, ou seja, a estratégia de ***panic mode***
- Em analisadores *bottom-up* isto se obtém acrescentando à gramática regras de tratamento de erros

exp : LPAR error RPAR ()

exps : exp ()
| error ; exp ()

exp : NUM ()
| exp PLUS exp ()
| LPAR exp RPAR ()

exps : exp ()
| exps ; exp ()

- Uma estratégia geral, válida tanto para análise *bottom-up* como para *top-down*, é a busca de um átomo de sincronização, ou seja, a estratégia de ***panic mode***
- Em analisadores *bottom-up* isto se obtém acrescentando à gramática regras de tratamento de erros

exp : LPAR error RPAR ()

exps : exp ()
| error ; exp ()

- Após um erro, busca-se um átomo de sincronização. Passos de recuperação:
  - Desempilhar (se necessário) até atingir um estado em que a ação para o átomo incorreto seja de *shift*
  - Executar um shift do átomo incorreto
  - Descartar símbolos de entrada (se necessário) até atingir um estado cuja ação associada não seja de erro.
  - Retomar a análise normal

```
exp : NUM                ()        exps : exp              ()
    | exp PLUS exp       ()             | exps ; exp       ()
    | ( exp )            ()
```

```
exp : ( error )          ()
exps : exp               ()
     | error ; exp       ()
```

Ainda a ser lido

entrada: **NUM PLUS (NUM PLUS @#$ PLUS NUM) PLUS NUM**

pilha: **exp PLUS (exp PLUS**

Foi encontrado um **@#$** , que é um átomo não esperado!

```
exp : NUM                 ()
    | exp PLUS exp        ()
    | ( exp )             ()
```

```
exps : exp                ()
     | exps ; exp         ()
```

```
exp : ( error )           ()

exps : exp                ()
     | error ; exp        ()
```

Ainda a ser lido

entrada: **NUM PLUS (NUM PLUS @#$ PLUS NUM) PLUS NUM**

pilha: **exp PLUS (**

Desempilhar até que uma análise correta resulte do ***shift*** de "**error**"

```
exp : NUM                 ()        exps : exp              ()
    | exp PLUS exp        ()             | exps ; exp       ()
    | ( exp )             ()
```

```
exp : ( error )           ()
exps : exp                ()
     | error ; exp        ()
```

Ainda a ser lido

entrada: **NUM PLUS (NUM PLUS @#$ PLUS NUM) PLUS NUM**

pilha: **exp PLUS (error**

Aplicar a ação ***shift*** "**error**"

```
exp : NUM                 ()
    | exp PLUS exp        ()
    | ( exp )             ()

exps : exp                ()
     | exps ; exp         ()
```

```
exp : ( error )           ()

exps : exp                ()
     | error ; exp        ()
```

Ainda a ser lido

emtrada: **NUM PLUS (NUM PLUS @#$ PLUS NUM) PLUS NUM**

pilha: **exp PLUS (error**

Descartar símbolos de entrada até que se possa aplicar uma ação válida de ***shift*** ou ***reduce***

exp : NUM ()
| exp PLUS exp ()
| (exp) ()

exps : exp ()
| exps ; exp ()

exp : (error) ()

exps : exp ()
| error ; exp ()

Ainda a ser lido

entrada: **NUM PLUS (NUM PLUS @#$ PLUS NUM) PLUS NUM**

pilha: **exp PLUS (error)**

Aplicar a ação ***Shift*** ao símbolo "**)**"

```
exp : NUM                 ()
    | exp PLUS exp        ()
    | ( exp )             ()
```

```
exps : exp                ()
     | exps ; exp         ()
```

```
exp : ( error )           ()

exps : exp                ()
     | error ; exp        ()
```

Ainda a ser lido

entrada: **NUM PLUS (NUM PLUS @#$ PLUS NUM) PLUS NUM**

pilha: **exp PLUS exp**

Aplicar ação ***reduce*** usando a regra **exp ::= (error)**

```
exp : NUM                 ()
    | exp PLUS exp        ()
    | ( exp )             ()
```

```
exps : exp                ()
     | exps ; exp         ()
```

```
exp : ( error )           ()

exps : exp                ()
     | error ; exp        ()
```

Ainda a ser lido

entrada: **NUM PLUS (NUM PLUS @#$ PLUS NUM) PLUS NUM**

pilha: **exp PLUS exp**

Agora a situação de erro foi recuperada, e já se pode continuar a análise…