

Alocação Local de Registradores

Geração de Código

Hervé Yviquel

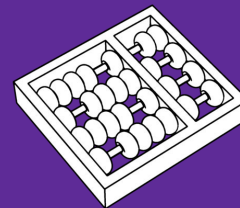
herve@ic.unicamp.br

Universidade Estadual de Campinas (Unicamp)

Instituto de Computação (IC)

Laboratório de Sistemas de Computação (LSC)

MC921 • Projeto e Construção de Compiladores • 2022 S2



UNICAMP

Aula Anterior

Plano

- Seleção de Instruções
- Arquitetura
- Primeiro exemplo
- Maximal Munch
- Custo da Cobertura
- Programação Dinâmica
- Geração de Código
- Eficiência

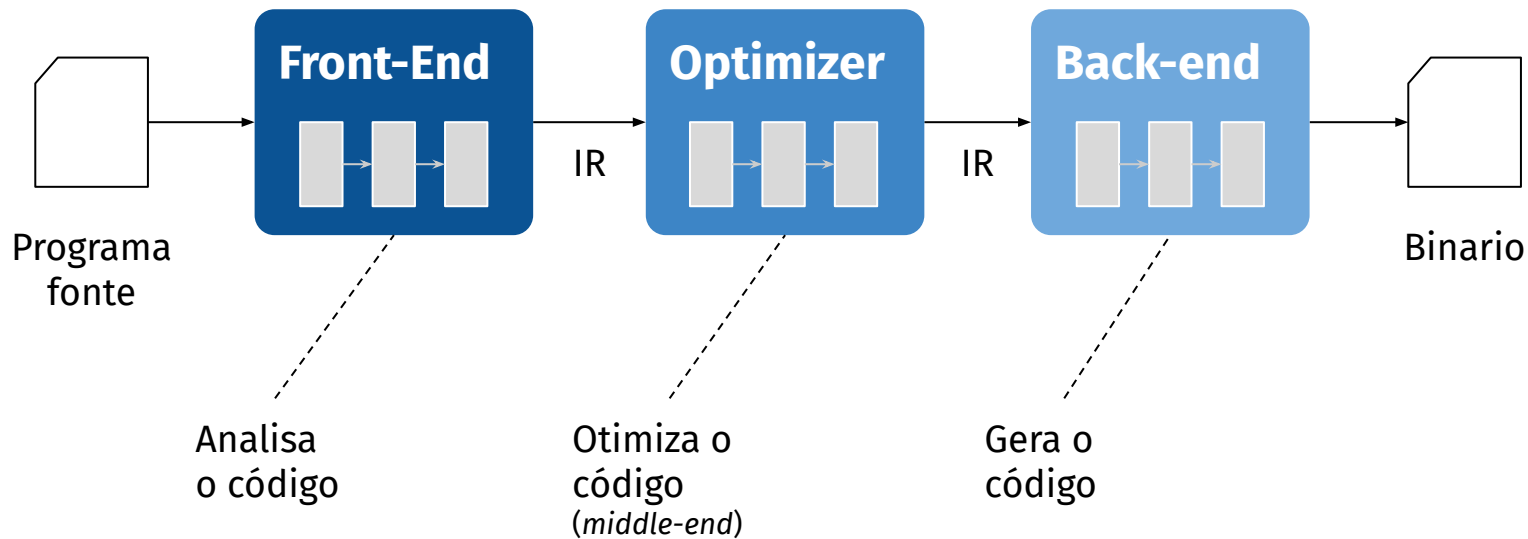
Aula de Hoje

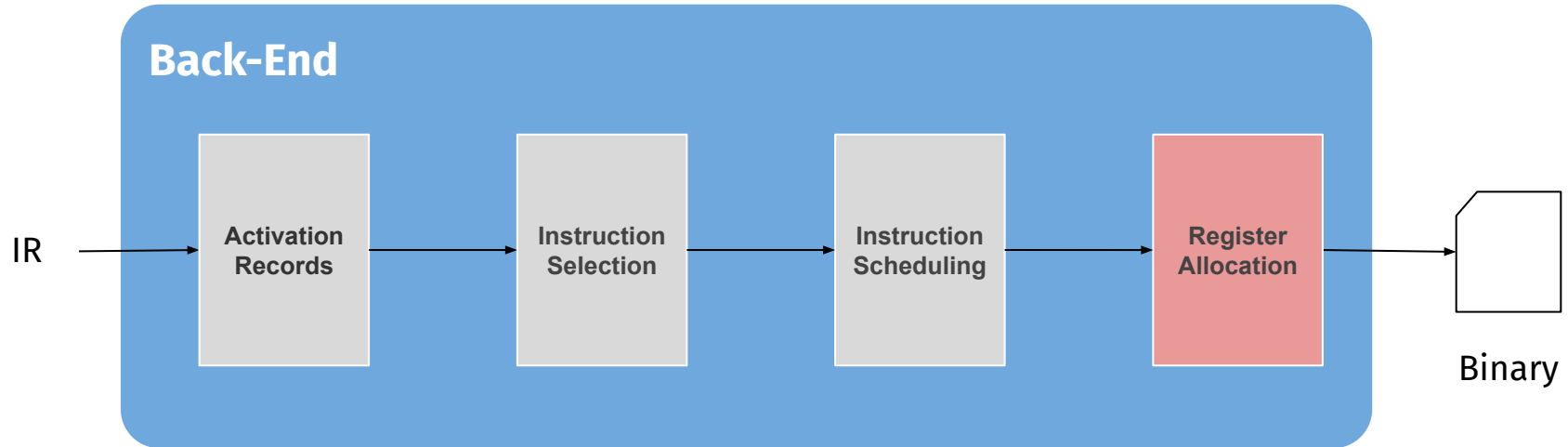
Resumo

- Introdução
- Arquitetura e Memória
- Alocação de Registro
- Primeiro Exemplo
- Algoritmo de Sethi-Ullman

Introdução



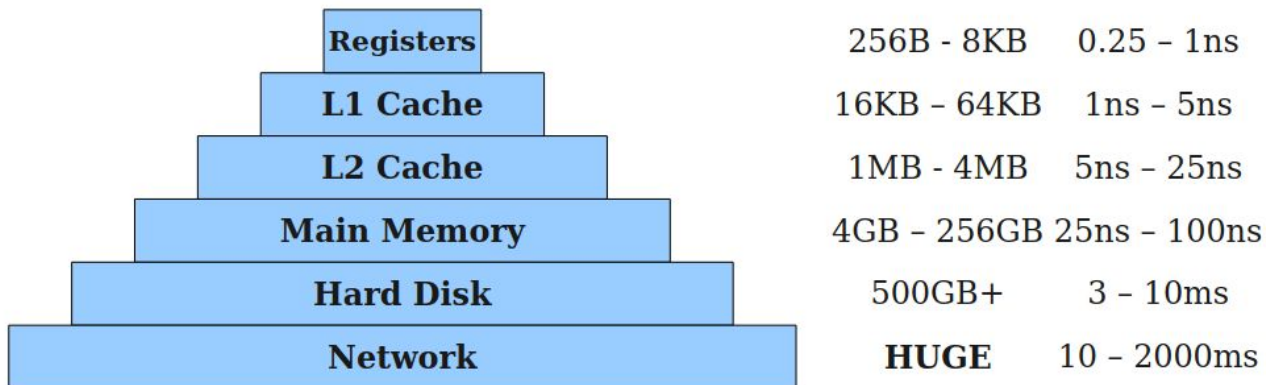




Arquitetura e Memória



- *Tradeoff* entre velocidade e tamanho na memória
 - Memórias rápida são caras (Registro, RAM, etc)
 - Memórias lenta são baratas (HD, SSD, etc)
- Ideia
 - Tente obter o melhor de todos os mundos usando vários tipos de memória



Número de Registradores em Arquiteturas Comuns

Arquitetura	32 bits	64 bits
ARM	15	31
Intel x86	8	16
MIPS	32	32
POWER/PowerPC	32	32
RISC-V	16/32	32
SPARC	31	31

ARM v8 Registers

10



...



Word size	Size (bits)	Name
Byte	8	Bn
Halfword	16	Hn
Word	32	Sn
Doubleword	64	Dn
Quadword	128	Qn

Alocação de Registradores

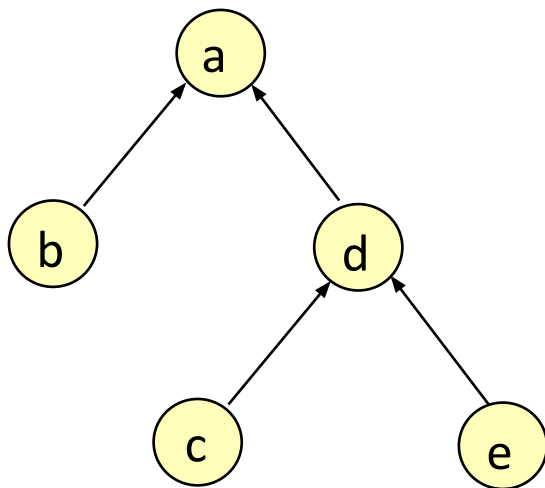


- A alocação global de registradores é muito cara
 - Usa coloração de grafos
 - Pode ser problemático para interpretadores e compiladores JIT
- Existem outras abordagens que alocam registradores dentro de um bloco básico
 - Chamado de alocação local de registradores
 - Por exemplo, o algoritmo de Sethi-Ullman (1970)

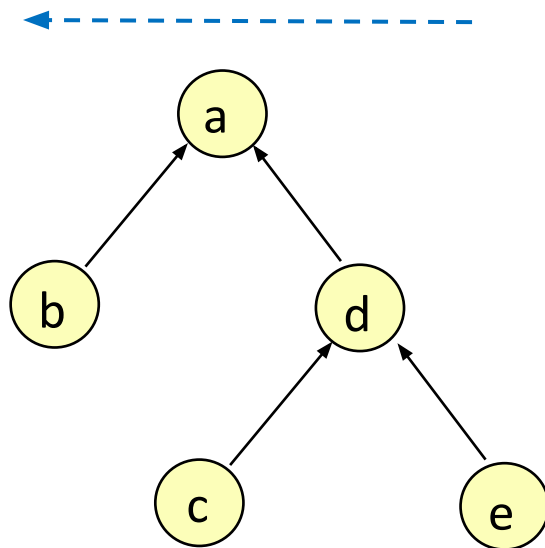
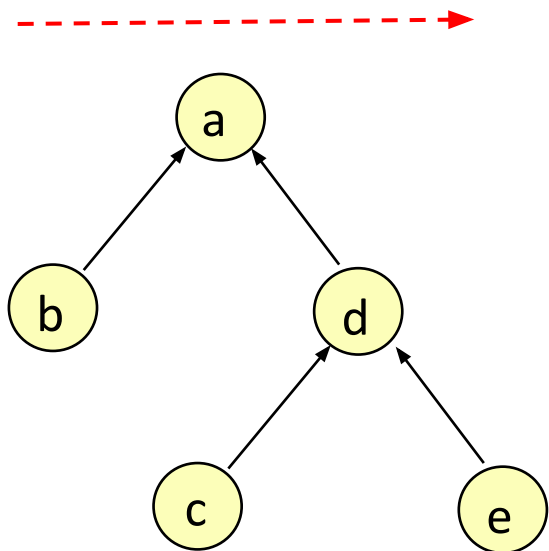
Primeiro Exemplo



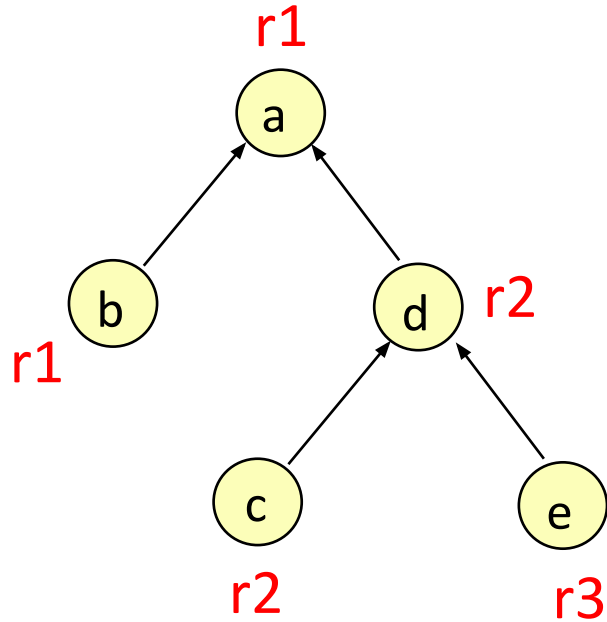
- Build the trees of a basic block
- Use scheduling to minimize the number of registers required by each tree
- Leaves are load operations into registers or have already been assigned to registers



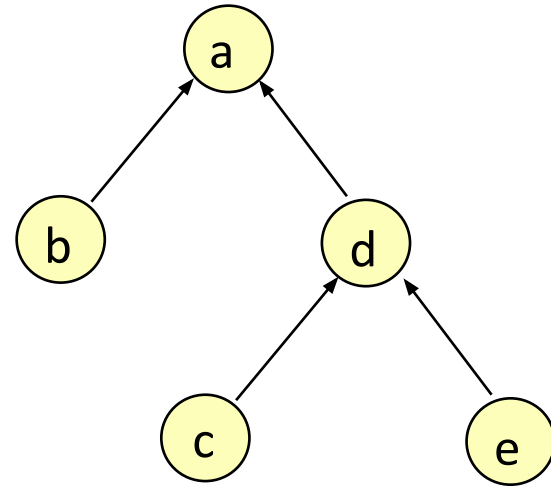
- Vamos considerar dois escalonamentos diferentes
 - e calcular o número de registradores necessários



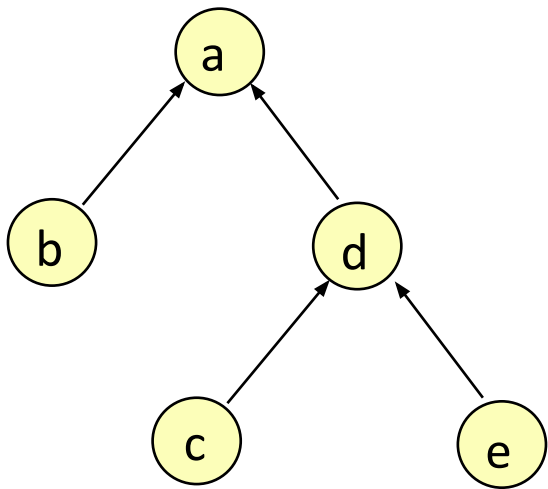
- Da esquerda para direita



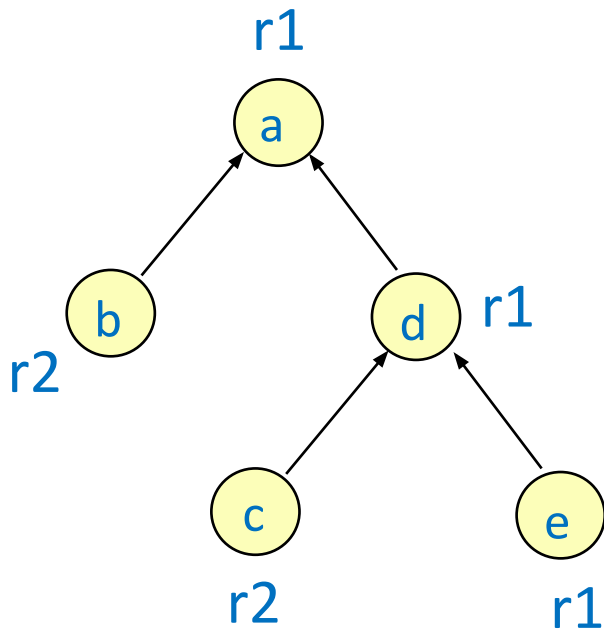
3 registers



- Da direita para esquerda



3 registers



2 registers

Algoritmo do Sethi-Ullman

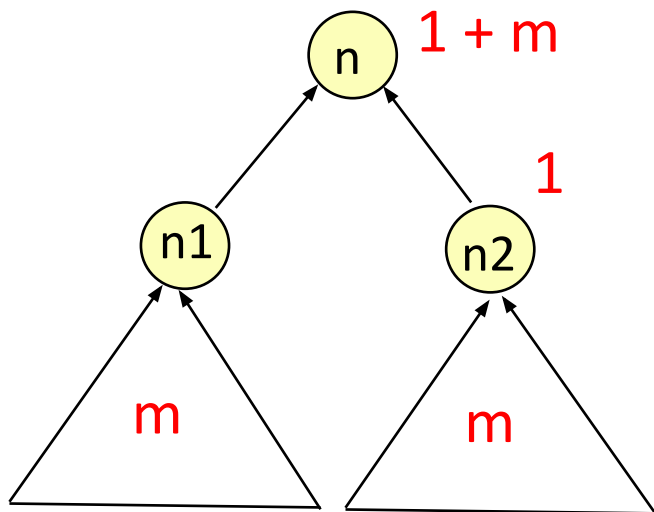


1. Etapa *Bottom-up*
 - Calcular número SU
2. Etapa *Top-down*
 - Selezione o escalonamento

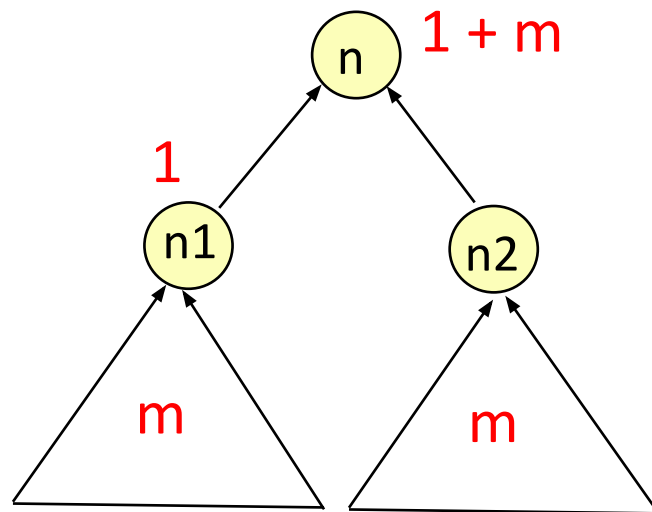
- O número de Sethi-Ullman de um nó n
 - que tem filhos(n) chamados n_i , $i = 1, 2$

$$SU(n) = \begin{cases} 1, & \text{se } n \text{ é uma folha} \\ 1 + m, & \text{se } m = m_1 = m_2, m_i = SU(n_i) \\ \max(m_i), & \text{se } m_1 \neq m_2, m_i = SU(n_i) \end{cases}$$

$$SU(n) = \begin{cases} 1, n \text{ é uma folha} \\ 1 + m, \text{ se } m = m_1 = m_2, m_i = SU(n_i) \\ \max(m_i), \text{ se } m_1 \neq m_2, m_i = SU(n_i) \end{cases} \quad \leftarrow$$

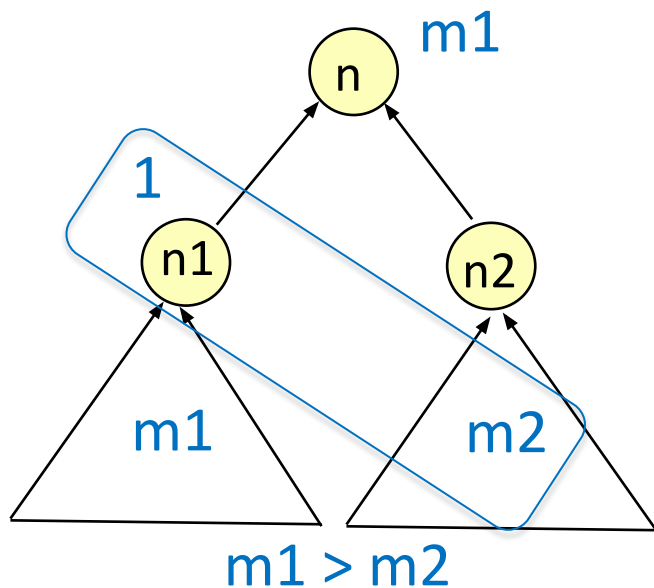


Right-to-Left



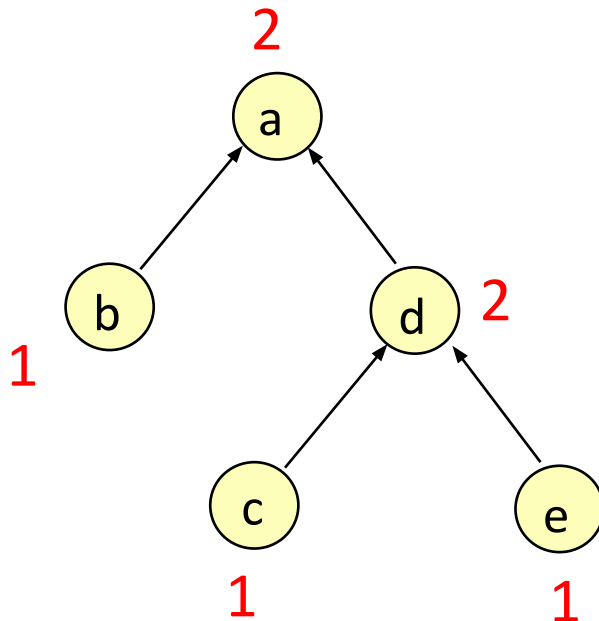
Left-to-Right

$$SU(n) = \begin{cases} 1, n \text{ é uma folha} \\ 1 + m, \text{ se } m = m_1 = m_2, m_i = SU(n_i) \\ \max(m_i), \text{ se } m_1 \neq m_2, m_i = SU(n_i) \end{cases}$$



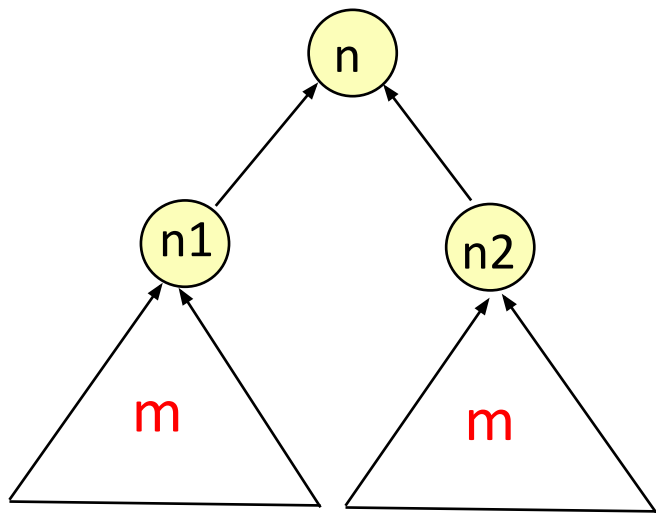
se $m_1 > m_2$,
 $m_1 \geq 1 + m_2$
é sempre verdadeiro

- Calcular números de Sethi-Ullman
 - Independente da ordem de travessia

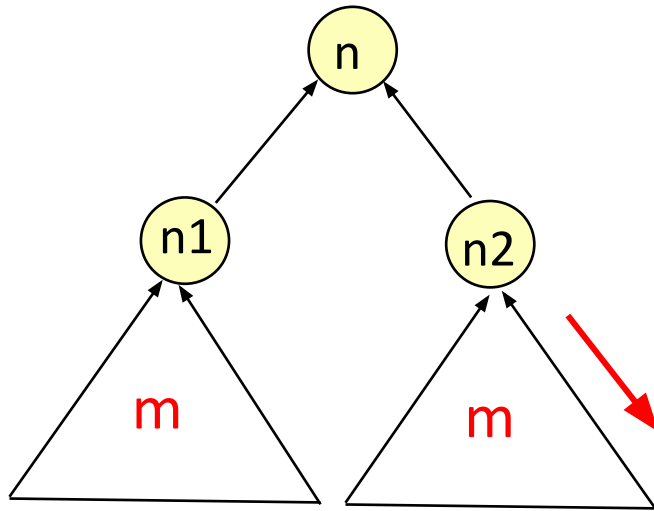


2 registers

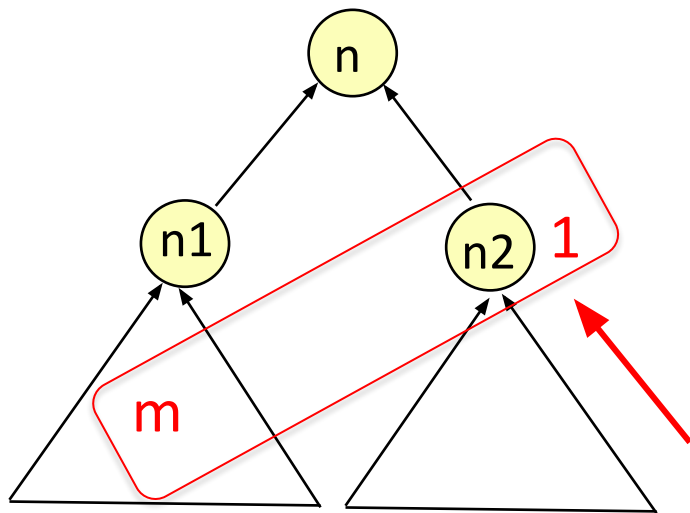
- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro



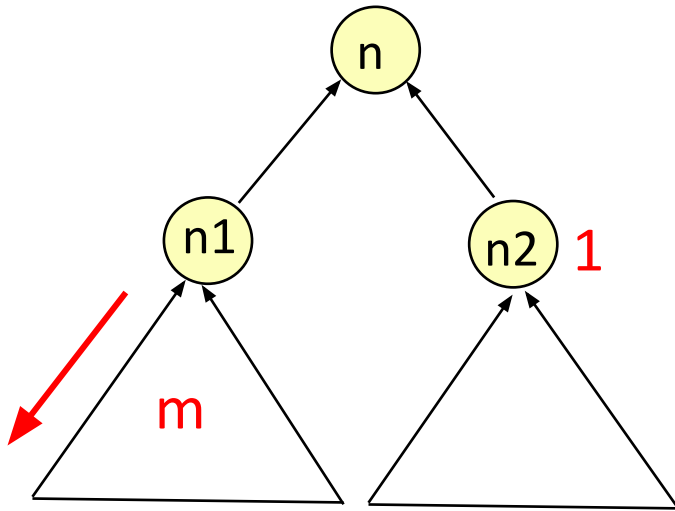
- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro



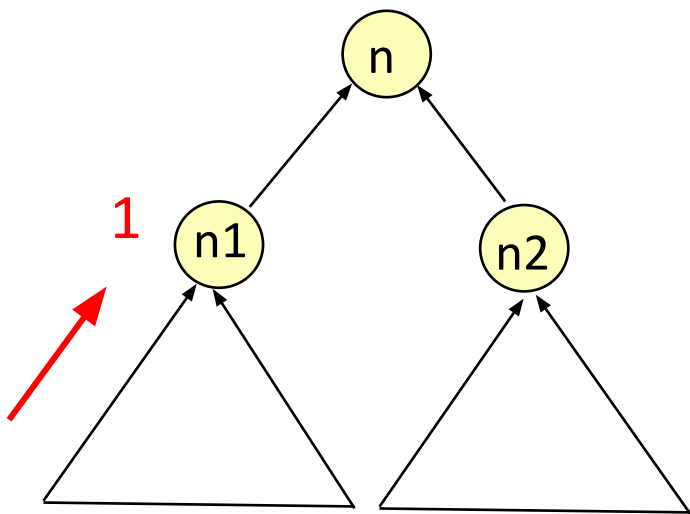
- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro



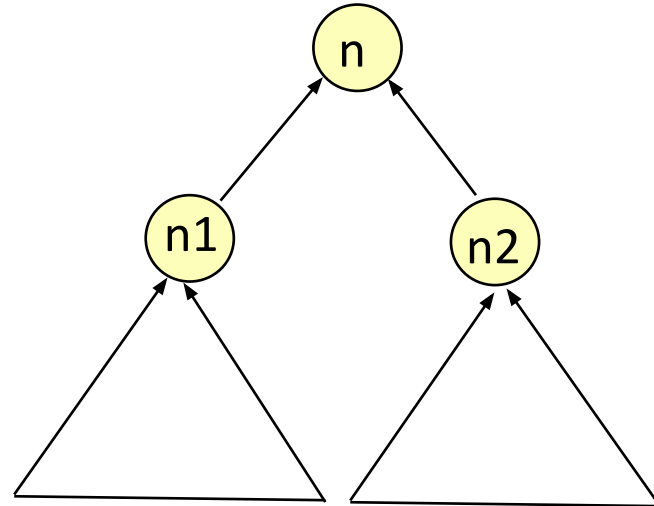
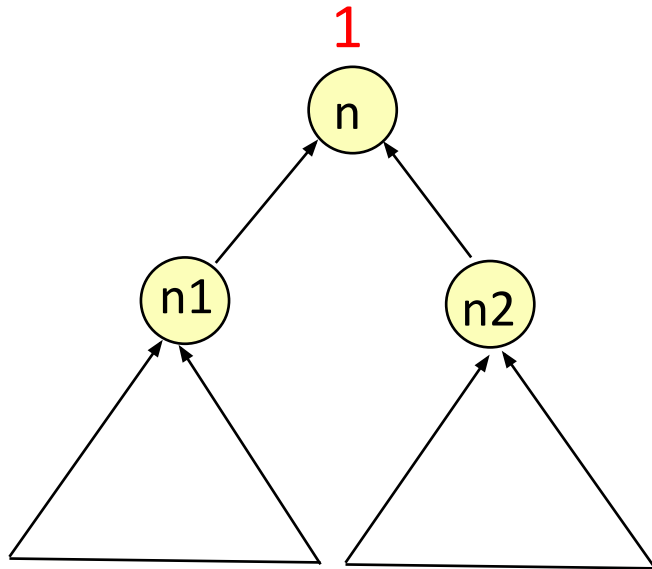
- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro




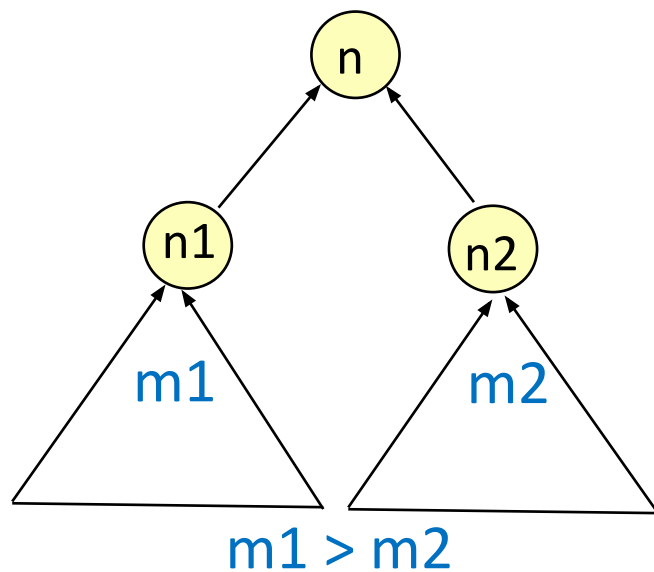
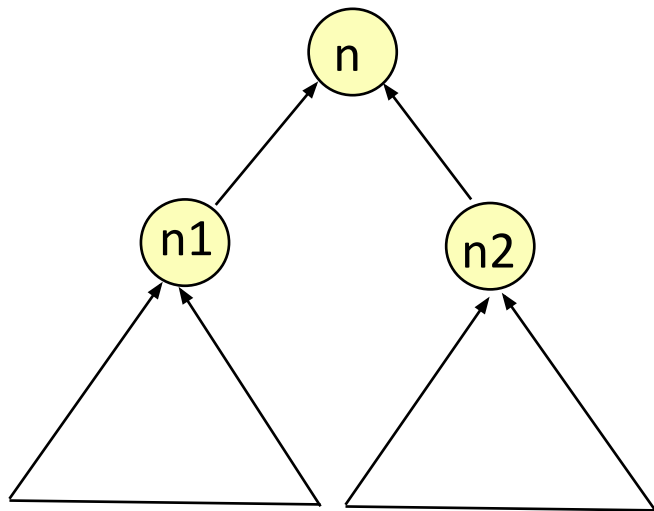
- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro



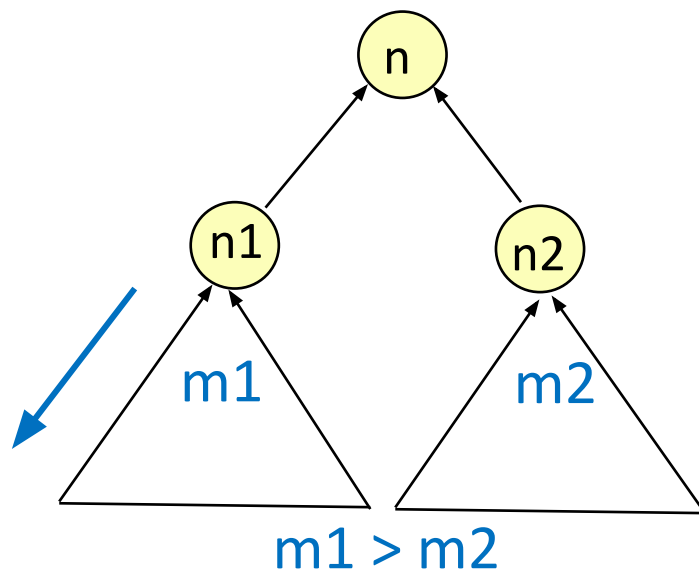
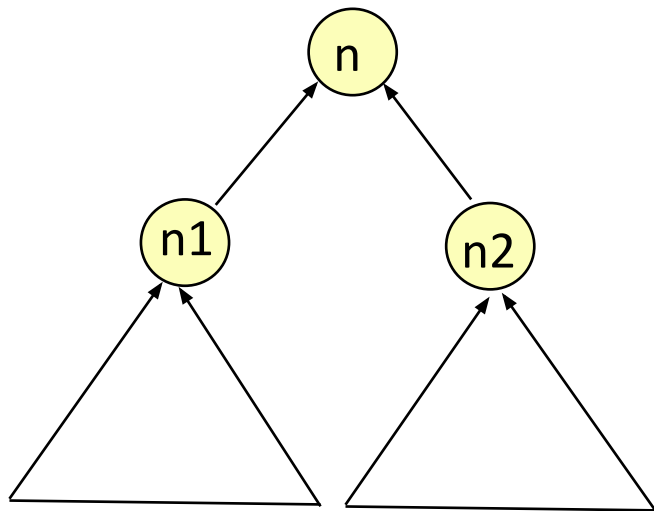
- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro



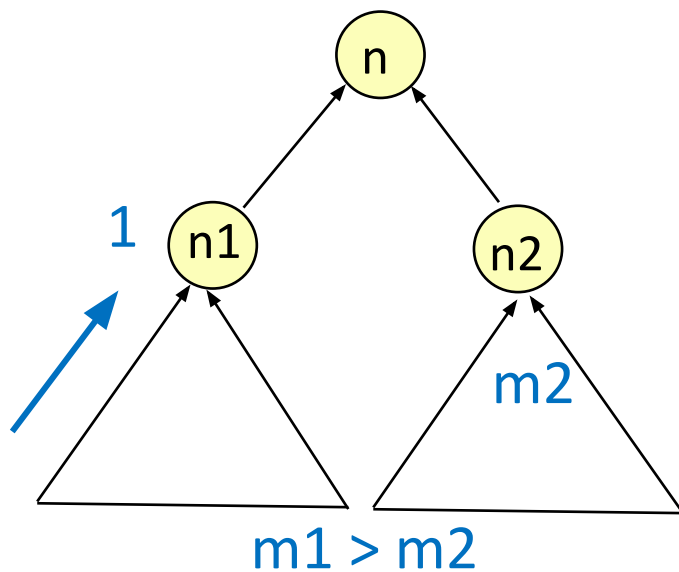
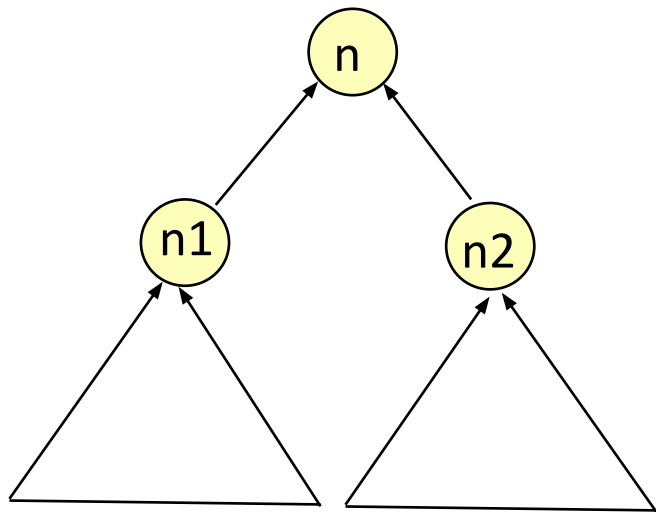
- Se $m = m1 = m2$, escolha qualquer filho para percorrer primeiro
- Se $m1 > m2$, escolha $n1$ para percorrer primeiro 




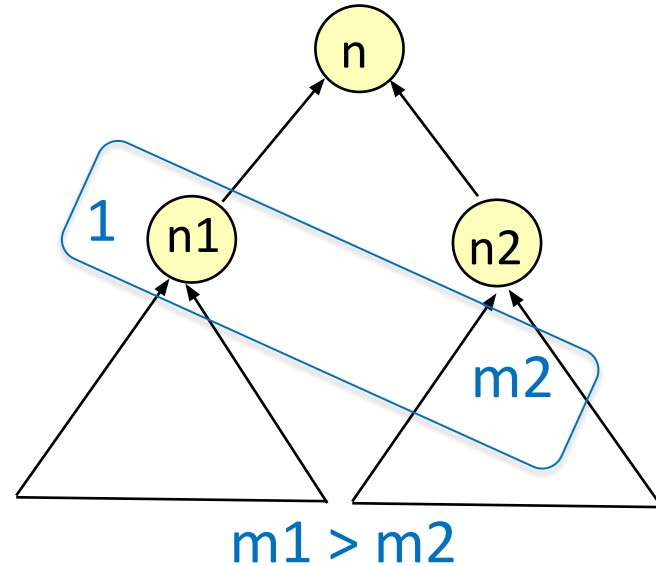
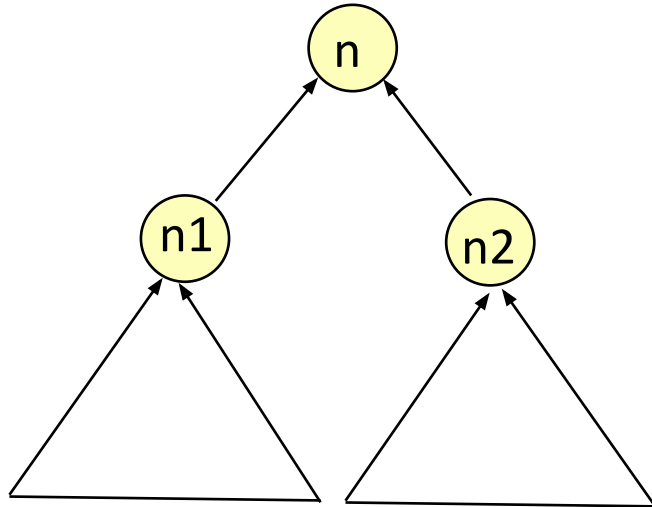
- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro ←



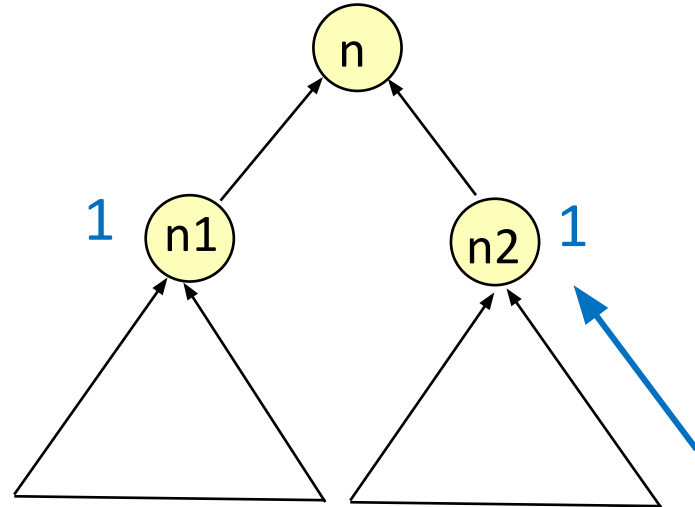
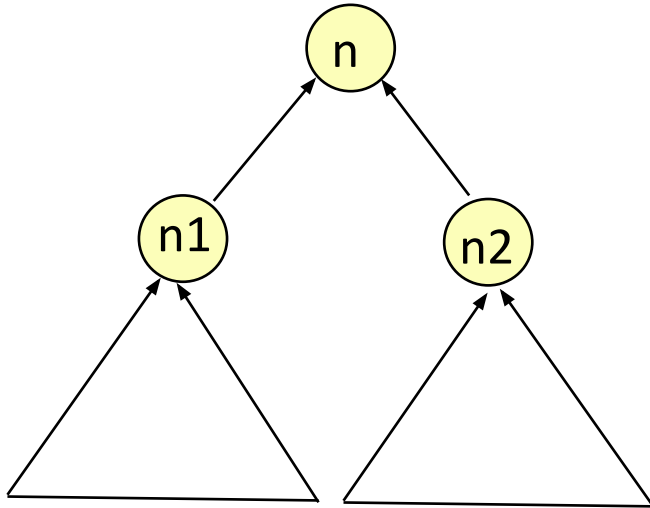
- Se $m = m1 = m2$, escolha qualquer filho para percorrer primeiro
- Se $m1 > m2$, escolha $n1$ para percorrer primeiro



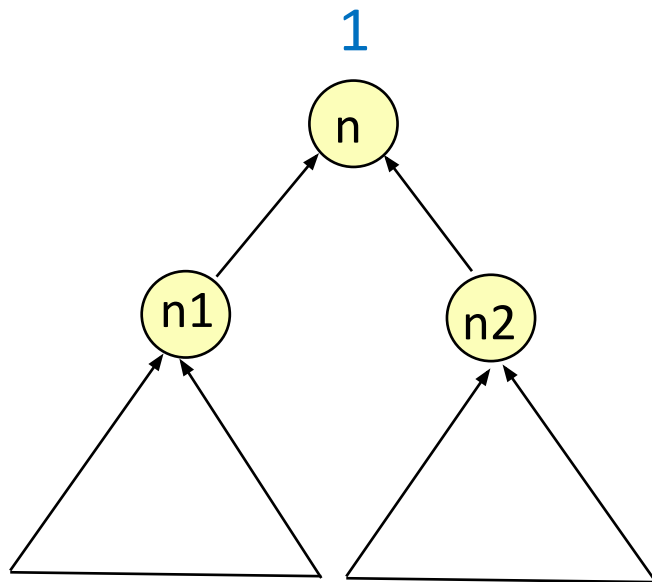
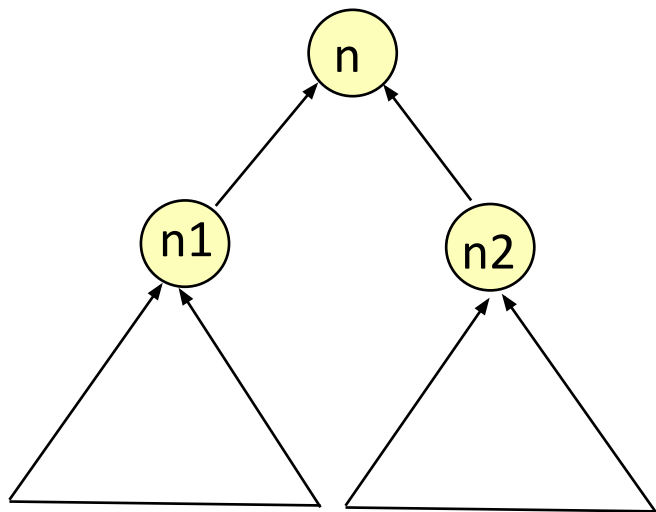
- Se $m = m1 = m2$, escolha qualquer filho para percorrer primeiro
- Se $m1 > m2$, escolha $n1$ para percorrer primeiro 



- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro ←

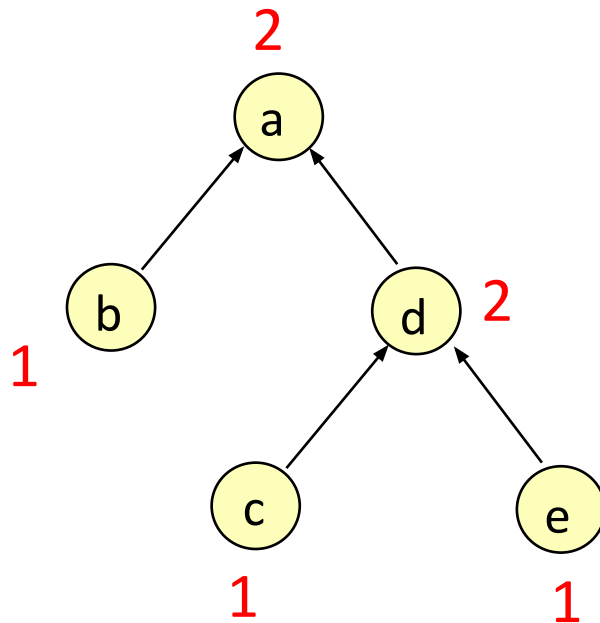


- Se $m = m_1 = m_2$, escolha qualquer filho para percorrer primeiro
- Se $m_1 > m_2$, escolha n_1 para percorrer primeiro



- Top-down pass: compute schedule

instr[c]
instr[e]
instr[d]
instr[b]
instr[a]



- Considere o seguinte bloco de μc em SSA
- Represente o bloco na forma de árvore
- Qual o mínimo de registradores de máquina que podemos usar neste bloco?
 - Utilize o algoritmo de Sethi-Ullman

bloco:

```
%1 = load int %x  
%2 = load int %y  
%3 = add int %1 %2  
%4 = literal int 42  
%5 = sub int %4 %2  
%6 = add int %3 %5  
%7 = add int %6 %1
```

Resumo

- Arquitetura e Memória
- Alocação de Registro
- Algoritmo de Sethi-Ullman

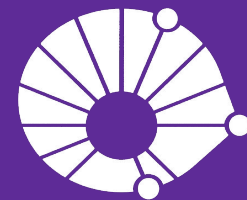
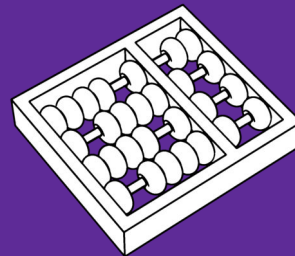
Leitura Recomendada

- Capítulo XX do livro do Appel
- Capítulo XX do livro do Cooper

Próxima Aula

- Análise de fluxo de dado

Obrigado!
Merci!



UNICAMP

Pallete



BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

BUBBLE

Lorem ipsum dolor
sit amet, consectetur
adipiscing elit.

DRACULA

Table Title	
Column 1	Column 2
One	Two
Three	Four

Table Title	
Column 1	Column 2
One	Two
Three	Four

Table Title	
Column 1	Column 2
One	Two
Three	Four

Table Title	
Column 1	Column 2
One	Two
Three	Four

