Primeira Prova de Análise Estática de Programas
- DCC888 -
Ciência da Computação

Nome: _____
"Eu dou minha palavra de honra que não trapacearei neste exame."

Número de matrícula: _____

As regras do jogo:

- A prova é sem consulta.
- Quando terminar, não entregue nada além do caderno de provas para o instrutor.
- Quando escrever código, a sintaxe correta é importante.
- Cada estudante tem direito a fazer uma pergunta ao instrutor durante a prova. Traga o caderno de provas quando vier à mesa do instrutor.
- A prova termina uma hora e quarenta minutos após seu início.
- Seja honesto e lembre-se: **você deu sua palavra de honra**.

Alguns conselhos:

- Escreva sempre algo nas questões, a fim de ganhar algum crédito parcial.
- Se não entender a questão, e já tiver gasto sua pergunta, escreva a sua interpretação da questão junto à resposta.
- A prova não é difícil, ela é divertida, então aproveite!

Tabela 1: Pontos acumulados (para uso do instrutor)

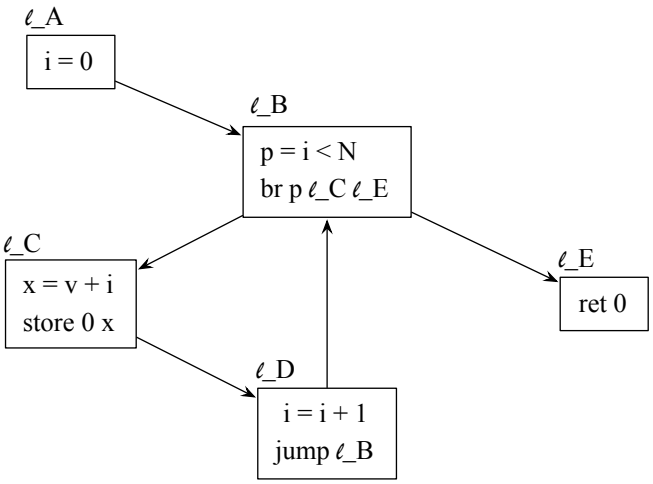
Questão 1	Questão 2	Questão 3	Extra

Questão Extra (0.5 Pontos): que filme ganhou o Oscar na categoria *Best Picture* em 2015?

1. Programas de computador podem gastar mais ou menos energia, dependendo de quais instruções eles usam. O objetivo desta questão é projetar uma análise estática que aproxima a maior potência dissipada por um programa. Assumiremos que a potência dissipada por um programa, em um dado instante de sua execução, é dada pela potência de cada uma das instruções no *pipeline* do processador. A sua análise estática deverá funcionar sobre uma linguagem com as seguintes instruções:

Produção		Símbolo		Potência(mW)
P	::=	$I; P$	; Programa	-
	|	ϵ	; Palavra vazia	-
I	::=	store v v	; Store	8
	|	load v v	; Load	10
	|	$v = v$	; Copy	1
	|	$v = v + v$	; Addition	3
	|	$v = v \times v$	; Multiplication	5
	|	$v = v < v$	; Greater than	2
	|	ret	; Unconditional jump	1
	|	jmp ℓ	; Unconditional jump	1
	|	br p ℓ ℓ	; Conditional jump	2
	|	noop	; No-operation	0

(a) (2 Pontos) Assumindo-se um processador de *pipeline* de três estágios, qual e a maior potência dissipada pelo programa abaixo? Para responder essa pergunta, considere que a potência dissipada por cada instrução é dada em *milliwatts*, de acordo com a tabela acima. Assim, você deve considerar qual é a sequência de três instruções que podem estar juntas no *pipeline* do processador – em qualquer ponto da execução do programa – que dissipa a maior potência.



(b) **[Continuação – Questão 1]** Uma solução para este problema usa um semi-reticulado bem simples, formado por triplas de instruções e números inteiros dentro de um certo intervalo:

$$L = \langle (I^3 \times [\text{low} \ldots \text{high}]), \leq, \vee, \bot, \top \rangle$$

Este semi-reticulado indica, para cada ponto do programa, qual é a maior potência que pode ser dissipada naquele ponto, e quais as instruções responsáveis por essa dissipação. Por exemplo, um elemento do conjunto subjacente pode ser:

$$\langle i = 0; p = i < N; \text{br } p\ \ell_C\ \ell_E \rangle \times 5$$

i. (1 Pontos) Qual o valor de "low"no reticulado? Trata-se da menor potência que pode ser dissipada em qualquer ponto do programa.

ii. (1 Pontos) Qual o valor de "high"no reticulado? Trata-se da maior potência que pode ser dissipada em qualquer ponto do programa.

iii. (2 Pontos) Quem é o supremum deste semi-reticulado ($\top$)?

iv. (2 Pontos) Quem é o ínfimum deste semi-reticulado ($\bot$)?

v. (2 Pontos) Como é definida a ordenação parcial ($<$) deste reticulado?

2. Nesta questão você deve criar funções de transferência para cada uma das dez instruções vistas na questão anterior. Suas funções de transferência precisam implementar uma análise "para-frente", que utiliza conjuntos IN e OUT a fim de calcular a maior potência dissipada pelo programa. Note que é necessário lembrar qual a sequência de instruções está-se considerando: por isso cada elemento do reticulado está associado a até três instruções. Cada questão vale um ponto.

   (a) store v v

   (b) load v v

   (c) $v = v$

   (d) $v = v + v$

   (e) $v = v \times v$

   (f) $v = v < v$

   (g) ret

   (h) jmp ℓ

   (i) br p ℓ ℓ

   (j) noop

3. Vamos agora dar os toques finais em nossa análise, definindo-se os operadores de junção, e a operação de inicialização.

   (a) (5 Pontos) Como deve ser inicializado o conjunto IN da primeira instrução do programa? Em nosso exemplo (visto na Questão 1), esse trata-se do conjunto IN associado a instrução "i = 0", no bloco ℓ_A.

   (b) (5 Pontos) Qual é o operador de junção (*least-upper bound*) do reticulado? Em outras palavras, como a informação é combinada nos pontos de junção do programa? Defina uma equação que faça a junção da informação.