Курсов проект за Машини с неограничени регистри

Калин Георгиев
kalin@fmi.uni-sofia.bg

27 октомври 2019 г.

1 Дефиниция на МНР и примери

Дефиницията на Машина с неограничени регистри по-долу е взаимствана от учебника [1] *А. Дичев, И. Сосков, "Теория на програмите", Издателство на СУ, София, 1998.*

"Машина с неограничени регистри" (или МНР) наричаме абстрактна машина, разполагаща с неограничена памет. Паметта на машината се представя с безкрайна редица от естествени числа $m[0], m[1], \ldots$, където $m[i] \in \mathcal{N}$. Елементите $m[i]$ на редицата наричаме "клетки" на паметта на машината, а числото i наричаме "адрес" на клетката $m[i]$.

МНР разполага с набор от инструкции за работа с паметта. Всяка инструкция получава един или повече параметри (операнди) и може да предизвика промяна в стойността на някоя от клетките на паметта. Инструкциите на МНР за работа с паметта са:

1) `ZERO n`: Записва стойността 0 в клетката с адрес n
2) `INC n`: Увеличава с единица стойността, записана в клетката с адрес n
3) `MOVE x y`: Присвоява на клетката с адрес y стойността на клетката с адрес x

"Програма" за МНР наричаме всяка последователност от инструкции на МНР и съответните им операнди. Всяка инструкция от програмата индексираме с поредния ѝ номер. Изпълнението на програмата започва от първата инструкция и преминава през всички инструкции последователно,

освен в някои случаи, описани по-долу. Изпълнението на програмата се прекратява след изпълнението на последната ѝ инструкция. Например, след изпълнението на следната програма:

```
0: ZERO 0
1: ZERO 1
2: ZERO 2
3: INC 1
4: INC 2
5: INC 2
```

Първите три клетки на машината ще имат стойност 0, 1, 2, независимо от началните им стойности.

Освен инструкциите за работа с паметта, МНР притежават и една инструкция за промяна на последователността на изпълнение на програмата:

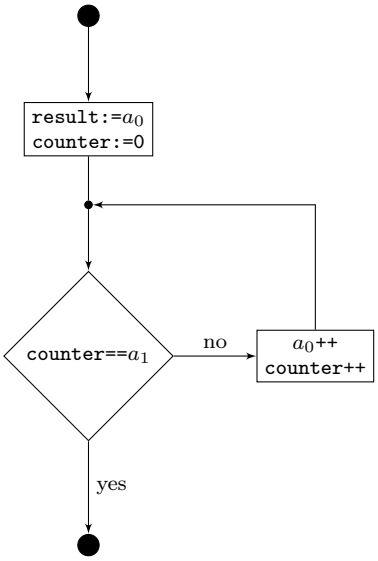
4) `JUMP z`: Изпълнението на програмата "прескача" и продължава от инструкцията с пореден номер z. Ако програмата има по-малко от $z+1$ инструкции, изпълнението ѝ се прекратява

5) `JUMP x y z`: Ако съдържанията на клетките x и y съвпадат, изпълнението на програмата "прескача" и продължава от инструкцията с пореден номер z. В противен случай, програмата продължава със следващата инструкция. Ако програмата има по-малко от $z+1$ инструкции, изпълнението ѝ се прекратява. Забележете, че горният вариант на инструкцията `JUMP` с един операнд (за безусловен преход) е частен случай на инструкцията `JUMP` с три операнда (за условен преход). Така `JUMP z` и може да се "симулира", например, с `JUMP 0 0 z`.

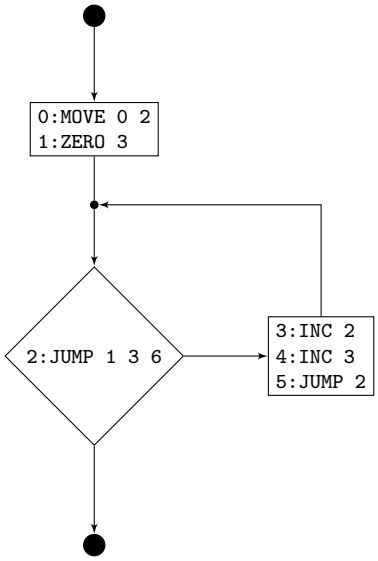
Например, нека изпълнението на следната програма започва при стойности на клетките на паметта $10, 0, 0, \ldots$:

```
0: JUMP 0 1 5
1: INC 1
2: INC 2
3: INC 2
4: JUMP 0
```

След приключване на програмата, първите три клетки на машината ще имат стойности 10, 10, 20.



(a) Програма за сумиране на числата a_0 и a_1 с използване само на операторите `=`, `==`, `++` и `if`.



(b) Програма за сумиране на клетките $m[0]$ и $m[1]$ с инструкции на МНР.

Фигура 1: Блок схеми на програма за сумиране на числа

Примери: На Фигура 1 (a) е показана блок схема на програма, използваща само операторите `=`, `==`, `++` и `if`, която намира в променливата `result` сумата на променливите a_0 и a_1. a_0 и a_1 считаме за дадени. Променливата `count` се инициализира с 0, а `result` - с a_0. В цикъл се добавя по една единица към `count` и `result` дотогава, докато `count` достигне стойността на a_1. По този начин, към `result` се добавят a_1 на брой единици, т.е. стойността ѝ се увеличава с a_1 спрямо налчалната ѝ стойност a_0.

На Фигура 1 (b) е показана същата програма, като операторите от първата са заменени със съответните им инструкции на МНР. Резултатът от програмата се получава в клетката $m[2]$, а за брояч се ползва клетката $m[3]$. На блок схемата са дадени поредните номера на инструкциите в окончателната програмата на МНР:

```
0:MOVE 0 2
1:ZERO 3
2:JUMP 1 3 6
3:INC 2
4:INC 3
5:JUMP 2
```

1.1 Примерни задачи за програми за МНР

1.1. Нека паметта на МНР е инициализирана с редицата $m, n, 0, 0, ...$ Да се напише програма на МНР, след изпълнението на която клетката с адрес 2 съдържа числото $m + n$.

1.2. Нека паметта на МНР е инициализирана с редицата $m, n, 0, 0, ...$ Да се напише програма на МНР, след изпълнението на която клетката с адрес 2 съдържа числото $m \times n$.

1.3. Нека паметта на МНР е инициализирана с редицата $m, n, 0, 0, ...$ Да се напише програма на МНР, след изпълнението на която клетката с адрес 2 съдържа числото 1 тогава и само тогава, когато $m > n$ и числото 0 във всички останали случаи.

2 Условие на проекта

Да се реализира интерпретатор за програми на МНР. Интепретаторът да работи в диалогов режим като приема инструкции за МНР и команди от стандартния вход. Във всеки момент интерпретаторът поддържа в паметта "заредена програма", състояща се от всички въведени или заредени от файл инструкции и команди на интерпретатора. Инструкциите са номерирани с последователни естествени числа спрямо реда на въвеждането им. Командите нямат номера, но също са част от програмата и са подредени заедно с инструкциите по реда на въвеждането им.

Интерпретаторът да поддържа следните команди (команди, които не са инструкции на МНР, започват със символа "/"):

2.1. `/load <file name>`: Зарежда програма за МНР от текстов файл. Програмата може да съдържа инструкции за МНР и команди на интерпретатора. Програмата не се изпълнява при зареждането на файла.

2.2. `/zero x y`: Нулира клетките на паметта с адреси от `x` до `y`.

2.3. `/run`: Изпълнява заредената програмата, както и всички команди на интерпретатора от заредената програма. Инструкциите и командите се изпълняват в реда, в който са зададени в изходния файл.

2.4. `/call x`: Изпълнява заредената програмата, както и всички команди на интерпретатора от заредената програма, като започва изпълнението от инструкцията с пореден номер `x`, а не от началото. Инструкциите и командите се изпълняват в реда, в който са зададени в изходния файл.

2.5. `/mem x y`: Извежда на стандартния изход съдържанието на клетките с адреси от `x` до `y`.

2.6. `/set x y`: Променя на `y` съдържанието на клетката с адрес `x`.

2.7. `/add <file name>`: Зарежда програма за МНР от текстов файл.

Обхват на програма наричаме наричаме целочисления интервал $[a, b]$, където a е най-малкия адрес на клетка от паметта, който се използва в някоя инструкция на интерпретатора (но не и в командите!), а b е най-големия такъв адрес.

Ако преди изпълнението на командата `/add` в паметта има заредена програма с обхват $[a, b]$, а новата програма има обхват $[a', b']$, новата програма да се преработи така, че да има обхват $[a' + b, b' + b]$. Инструкциите и командите на новата програмата да се добавят към края на заредената в паметта програма.

2.8. `/code`: Извежда на стандартния изход заредена в паметта програма (заедно с командите на интерпретатора в нея в съответния ред).

Внимание: За реализация на паметта на МНР да се използва "разреден масив" (*sparse array*) по алгоритъм, избран от вас.

Да се демонстрират командите `/add` и `/call` така, че една програма да ползва друга като подпрограма (например програма за умножение на числа да използва програма за събиране на числа като подпрограма).

Литература

[1] А. Дичев, И. Сосков, "Теория на програмите", Издателство на СУ, София, 1998