ПРИМЕНЕНИЕ ПРЕДМЕТНЫХ ЯЗЫКОВ ДЛЯ АВТОМАТИЗАЦИИ ВЫСОКОПРОИЗВОДИТЕЛЬНЫХ ВЫЧИСЛЕНИЙ

ВОСТОКИН С.В.

САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Цель и метод работы

- **Цель:** автоматизировать применение высокопроизводительной техники
  - **требования к пользователю:** умение описать свой алгоритм на процедурном языке высокого уровня; понимание базовых концепций такого языка
- **Метод:** применение предметных языков, указывающих исполняющей системе, как выполнить распараллеливание

Что понимается под предметным языком?

- Языки общего назначения высокого уровня:
  - C/C++, Java, C#
- Предметные языки автоматизации численного моделирования:
  - MATLAB (векторные вычисления);
  - Wolfram Language (комбинация символьной и численной математики);
  - Maple (символьные вычисления);
  - Mathcad (интерактивные вычисления, визуализация) и др.

Проблема

- **Преимущество** предметных языков – синтаксис и семантика адаптированы к традиционной математической нотации, что существенно снижает трудоёмкость программирования
- **Недостаток** – если требуется оптимизация кода, предметные языки уступают языкам общего назначения высокого уровня C/C++
- **Вывод:** нужно комбинировать эти два подхода

Как выполняется комбинирование?

- Концепция **скелета программы**: проект на языке программирования высокого уровня (C/C++), который, в отличие от проекта программы на псевдокоде, компилируется
- Концепция «внешней» **аннотации скелета** программы на предметном языке

Предлагаемые предметные языки

- организации вычислений в парадигме «портфель задач»;
- описания акторной модели вычислений, реализованной в общей и распределённой памяти;
- описания ввода данных;
- описания вывода данных;
- вызова функций стандартных математических библиотек;
- описания серий экспериментов

Пример скелета программы (1/3)

```
const int N=10;
double a[N][N],b[N][N],c[N][N];

struct Result{
  void save(ostream&s){ s<<num;
   for(int j=0;j<N;j++)s<<c[num][j]; }
  void rest(istream&s){ s>>num;
   for(int j=0;j<N;j++)s>>c[num][j]; }
    int num;// номер вычисленной строки
};

struct Task{
    void save(ostream&s){ s<<num; }
    void rest(istream&s){ s>>num; }
    int num;// номер вычисляемой строки
};
```

Пример скелета программы (2/3)

```
void Proc(Task&t,Result&r){
    int i=t.num;// параллельное вычисление строки матрицы C
    for(int j=0;j<N;j++){
        c[i][j]=0.0;
        for(int k=0;k<N;k++)c[i][j]+=a[i][k]*b[k][j];
    }
    r.num=i;
}

struct Bag{
    Bag(int argc, char* argv[]);
    void run();
    bool isTask(){return cur<N;}
    void put(Result&){ }
    void get(Task&t){t.num=cur++;}
    int cur;//номер текущей строки в матрице C
};
```

Пример скелета программы (3/3)

```
int main(int argc, char* argv[])
{
    Bag bag(argc,argv);
    // инициализация
    input(a,b);
    bag.cur=0;
    // параллельное умножение матриц
    bag.run();
    // вывод результата параллельного умножения
    output(c);

    start_time();
    strassen(a,b,c); // теперь умножаем методом Штрассена
    end_time();
    return 0;
}
```

Примеры реализации представленного подхода

- Проект Templet
  - http://templet.ssau.ru
  -
- Язык разметки Templet
  - http://github.com/templet-language

Спасибо за внимание!

- Контакты:

  Востокин Сергей Владимирович

  easts@mail.ru