УДК 537.612.2

В.С. Гуров, А.А. Дягилев, В.С. Иванов, А.А. Трубицын

РЕАЛИЗАЦИЯ АЛГОРИТМА РАСПАРАЛЛЕЛИВАНИЯ ВЫЧИСЛЕНИЙ ПРИ РЕШЕНИИ ЗАДАЧ ТЕОРИИ ПОТЕНЦИАЛА МЕТОДОМ ГРАНИЧНЫХ ЭЛЕМЕНТОВ

Разработан и реализован алгоритм параллельных вычислений массива потенциалов в электронно-оптических системах методом граничных элементов. Проведены сравнительные характеристики разработанного алгоритма со стандартным, однопоточным. Показана возможность значительной экономии (более 40 %)временных ресурсов в случае двухпоточных вычислений.

Ключевые слова: *метод граничных элементов, моделирование электронно-оптических систем, многопоточные вычисления*

Введение. К одному из основных этапов компьютерного моделирования систем электронной оптики относится вычисление функции распределения потенциала в рабочем объеме. При этом функция распределения определяется в дискретном наборе заранее фиксированных узлов. Временные затраты на моделирование систем в целом, по сути, обусловлены объемом вычислений на данном этапе и напрямую зависят от степени дискретизации исследуемой области, т.е. от количества узловых точек, что при современных требованиях к точности вычислений предопределяет значительный объем результирующих (многомерных) массивов.

В настоящее время наиболее широкое распространение получили следующие численные методы решения задач теории потенциала – метод конечных разностей (МКР), метод конечных элементов (МКЭ) и активно развивающийся метод граничных элементов (МГЭ) [1,2]. Метод граничных элементов имеет ряд преимуществ по отношению к МКР и МКЭ, а именно возможность решения задач с практически произвольной конфигурацией границы, высокую точность вычислений и простоту алгоритмизации. Появление в арсенале исследователей многопроцессорной вычислительной техники выявило еще одно положительное качество МГЭ – возможность элементарного распараллеливания потоков данных и их независимую обработку, в то время как попытки создания надежных, легко адаптируемых к конкретным задачам параллельных алгоритмов для МКР и МКЭ оказываются мало успешными. Указанная проблема заключена в самой сути методов – в МКР и МКЭ вычисление потенциала в каждом конкретном узле сетки невозможно провести независимым образом от остальных узлов; в МГЭ потенциал может быть автономно рассчитан в любой точке пространства. С целью демонстрации последнего утверждения далее описывается суть метода граничных элементов.

Для численного решения задачи Дирихле интегральному уравнению, представляющему собой комбинацию потенциалов простого и двойного слоев [3], придается дискретная форма, для чего граница Γ области G разбивается на N граничных элементов. С учетом постоянства потенциала на каждом электроде и при предположении постоянства нормальной производной потенциала на каждом граничном элементе указанное уравнение записывается в виде

$$\gamma u(\xi)+\sum_{j=1}^{N} u_j H_j(\xi)=\sum_{j=1}^{N} q_j F_j(\xi), \qquad (1)$$

где $\xi \in G \cup \Gamma$; $\gamma=\pi$ при $\xi\in\Gamma$ и $\gamma=2\pi$ при $\xi\in$G для двумерных задач, $\gamma=2\pi$ при $\xi\in\Gamma$ и $\gamma=4\pi$ при $\xi\in$G для трехмерных задач, а функции $F_j(\xi)$ и $H_j(\xi)$ являются интегралами от фундаментального решения и от нормальной производной фундаментального решения уравнения Лапласа [1] соответственно.

Вычисление электростатического поля проводится в два этапа. Сначала с помощью уравнения (1) рассчитывается неизвестный вектор q_j по известному граничному распределению потенциала u ($\xi\in\Gamma$), т.е. решается "обратная" задача. Затем используются найденные значения q_j и заданные u_j для определения из уравнения (1) функции $u(\xi)$, $\xi\in\Omega$, т.е. решается "прямая" задача.

Решение прямой задачи, после того как решена обратная задача, базируется на очевидной формуле [см. уравнение (1)], позволяющей вы-

числить потенциал в любой точке пространства.

$$u(\xi)=\frac{1}{\gamma}\left[\sum_{j=1}^{N}u_j H_j(\xi)-\sum_{j=1}^{N}q_j F_j(\xi)\right],\ \xi\in\Omega. \quad (2)$$

Максимально краткая запись формулы (2), определяющей, к примеру, в двумерном случае потенциал в точке $\xi=\xi(x,y)$, выглядит следующим образом:

$$u=F(x,y), \quad (3)$$

где функция $F(x,y)$ есть правая часть выражения (2).

Последующие стадии моделирования систем электронной оптики требуют проведения траекторного анализа, где необходимо знание составляющих градиента потенциала. Составляющие градиента могут быть надежно, с удовлетворительной точностью и малыми временными затратами вычислены по конечно-разностным формулам [4], для использования которых предварительно должна быть создана равномерная сетка потенциалов.

Создание сетки потенциалов $u_{ij}=F(x_j, y_i)$ в прямоугольной области $x\in[Xmin,Xmax]$, $y\in[Ymin,Ymax]$ на основе отношения (3) осуществляется с помощью простейшего алгоритма, выраженного средствами языка Object PASCAL:

```
for i:=1 to nY do
begin
   Y:=Ymin+Hy*(i-1);
    for j:=1 to nX do
    begin
        X:=Xmin+Hx*(j-1);
         U[i,j]:=F(X,Y);
     end; // j
end; // i
```

Здесь *nX, nY* – количество узлов сетки по направлениям *0X* и *0Y*. Соответственно:

Hx = (Xmax-Xmin)/(nX-1),
Hy = (Ymax-Ymin)/(nY-1).

Наиболее трудоёмкой задачей в данном цикле является вычисление функции *F(X,Y)*. Для разделения процесса вычислений [5] выделим как независимую часть цикл по *j*. При запуске такого цикла с различными значениями *Y* получим уменьшение итераций цикла по *i*. Модифицированный алгоритм вычисления сетки потенциалов с помощью двух независимых потоков будет выглядеть следующим образом:

```
type
 TThreadDomain = class(TThread)
 protected
  procedure Execute; override;
 public
  U: array of Double;
  constructor Create(i, Y: Integer);
  destructor Destroy; override;
End;

Implementation

Procedure TThreadDomain.Execute;
// Выполнение потока.
var
 j: Integer;
 X: double;
Begin
 SetLength(U, nX+1);
 for j:=1 to nX do
 begin
  X:=Xmin+Hx*(j-1);
  U[j]:=F(X,Y); // Расчёт потенциалов
 end; // j
Event1.SetEvent; // Событие завершения потока
End; // Thread

// -modified process- //
var
 Event1: TEvent;
 Y: double;
 i, Imax, j, k: integer;
 Thr0, Thr1: TThreadDomain;
Begin
 Event1:=TEvent.Create(nil, False, False, '1');
 k:= (nY div 2);   // Остаток от деления
 Imax:= (nY mod 2); // Количество итераций по i

 for i:=1 to Imax do // Основной цикл вычислений
 begin
   Y:= Ymin + Hy*(2*i-2);
   Thr0:=TThreadDomain.Create(i, Y);
   Y:= Ymin + Hy*(2*i-1);
   Thr1:=TThreadDomain.Create(i, Y);
   Event1.WaitFor(90000); // Ждём первый поток
    Event1.WaitFor(90000); // Ждём второй поток
 For j := 1 to nX do // Присвоение вычисленных значений
 begin
  U[2*i-2, j]:=Thr0.U[j];
  U[2*i-1, j]:=Thr1.U[j];
 end; // j
end; // i

 For i:= 1 to k do // Дополнительный цикл вычислений
```

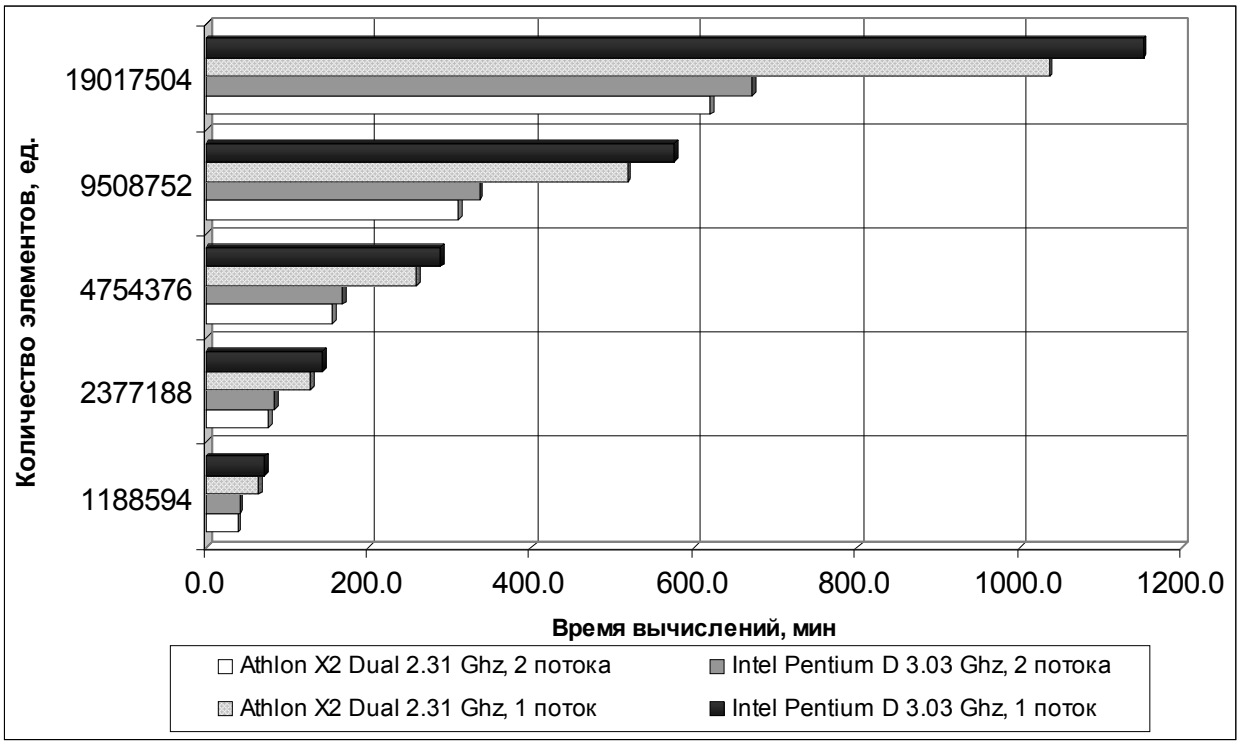

Рисунок – Зависимость времени счета от объема массива потенциалов

```
    Begin
      Y:= Ymin + Hy*(2*Imax+i);
      U[i, 2*Imax+i]:=F(X, Y);
    End; // i
  End;
```

В основном цикле происходит запуск двух потоков Thr0 и Thr1, затем метод WaitFor(90000) выделяет 90 с. на выполнение потоков. Таким образом, количество итераций уменьшается вдвое. В случае когда nY нечётно, в массиве $U[i,j]$ остаются невычисленными элементы при $i=nY$. Дополнительный цикл решает эту проблему. Алгоритм для произвольного количества потоков выглядит следующим образом:

```
  const
    M = 10;
  Var
    ThrArr: array [1..M] of TThreadDomain;
  begin
    k:= (nY div M);  // Остаток от деления
    Imax:= (nY mod M);  // Количество итераций по i

    for i:=1 to Imax do  // Основной цикл вычислений
    begin
      for j:=1 to M do  // Запуск M потоков вычислений
      begin
        Y:= Ymin + Hy*(M*i-j);
        ThrArr[j]:=TThreadDomain.Create(i, Y);
      End; // j
      for j:=1 to M do  // Ожидаем завершения потоков
        Event1.WaitFor(90000);
      for j:= 1 to M do      // Присвоение
       for k:= 1 to nX do   // вычисленных
        for l:= 1 to M do   // значений
          U[M*i-l, k]:=ThrArr[j].U[k];
    end; // i
  end;

    For i:= 1 to k do  // Дополнительный цикл вычислений
    Begin
      Y:= Ymin + Hy*(M*Imax+i);
      U[i, M*Imax+i] := F(X,Y);
    End; // i
```

Алгоритм отличается от предыдущего тем, что в основной цикл вычислений добавлен запуск M потоков вычислений (M – натуральное число).

С целью установления конкретного выигрыша в скорости вычислений были проведены исследования на наиболее распространенных в настоящее время процессорных системах. Рисунок демонстрирует временные затраты на расчет двумерной функции распределения потенциала для двух типов процессоров в зависимости от объема массива $nX \times nY$ в случае организации одно- и двухпоточных вычислений.

Анализ данных на рисунке позволяет сделать вывод о 40 % экономии временных затрат на решение задачи определения дискретной функции распределения потенциала, при использовании двухпроцессорных ЭВМ.

Заключение. Разработан алгоритм организации многопоточных вычислений функции распределения потенциала в электронно-оптических системах методом граничных элементов. Проведен сравнительный анализ временных затрат на решение задачи при использовании одно- и двухпоточных вычислений. Сделан вывод о значительной (>40 %) экономии временных ресурсов в последнем случае.

Библиографический список

1. *Бреббия К., Телес Ж., Вроубель Л.* Методы граничных элементов. – М.: Мир, 1987. – 524 с.

2. *Трубицын А.А.* Вычисление сингулярных интегралов при решении задачи Дирихле методом граничных элементов // Журнал вычислит. матем. и матем. физики. – 1995 – Т.35. № 4. – С. 532-541.

3. *Корн Г. и Корн Т.* Справочник по математике для научных работников и инженеров. – М.: Наука, 1974. – 832 с.

4. *Самарский А.А.* Введение в численные методы. – М.: Наука, 1987. – 256 с.

5. *Таненбаум Э.* Современные операционные системы. – СПб: Питер, 2002. – 1040 с.