|  |  |  |
| --- | --- | --- |
| ё | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Лабораторная работа № 1 | | |
| по дисциплине «Программирование вычислений» | | |
| **Алгоритмы и структуры данных для работы с разреженными матрицами** | | |
|  | | |
|  | Бригада 5 | Гриневич Юлия |
| Группа ПМ-21 | Егупов Иван |
| Вариант 9.1 | Порсин Данил |
|  |  |
|  |  |
|  |  |
| Преподаватели | Рояк Светлана Хаимовна |
|  |  |
| Новосибирск, 2024 | | |



1. **Цель работы**

Изучение псевдодинамического распределения памяти на языке ФОРТРАН; изучение форматов хранения матриц большой размерности; оптимизация программ по точности, скорости, памяти; изучение погрешности вычисления скалярного произведения; изучение способов отладки; изучение принципов формирования тестов для вычислительных программ; изучение файлов прямого доступа.

1. **Особенности программы**

Программа умножает симметричную матрицу на вектор. Матрица представлена в профильном формате. Хранится только нижний треугольник построчно. Для входных и выходных данных выделены файлы *di* (диагональные элементы), *al (*строки нижнего треугольника), *ia* (информация о профиле), *vector (*вектор, на который умножается матрица), *dim* (размерность матрицы), *res (*результат умножения, только текстовый формат). Элементы результирующего вектора представлены в экспоненциальном формате с точностью в четыре значащие цифры.

1. **Текст программы (матрица представлена в текстовых файлах)**

program main

implicit none

common/scal/dim,al\_size

common/lim/memlim

real mem

dimension mem(2\*\*28)

integer dim,al\_size

integer memlim

memlim = 2\*\*28

call input\_scal

call is\_mem\_enough

call input\_di(mem(1))

call input\_al(mem(dim+1))

call input\_ia(mem(dim+al\_size+1))

call input\_vect(mem(dim\*2+al\_size+2))

call func(mem(1),mem(dim+1),mem(dim+al\_size+1),

\*mem(dim\*2+al\_size+2),mem(dim\*3+al\_size+2))

call output(mem(dim\*4+2))

end

subroutine input\_scal

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

open(1, file='scal.txt', status='old')

read(1,\*)dim,al\_size

close(1)

end

subroutine is\_mem\_enough

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

if ((dim\*4+1+al\_size).gt.memlim) then

print\*,'not enough memory'

stop

end if

end

subroutine input\_di(di)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real di

dimension di(dim)

open(2, file='di.txt', status='old')

read(2, \*)(di(i), i=1,dim)

close(2)

end

subroutine input\_al(al)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real al

dimension al(al\_size)

if (al\_size.ne.0) then

open(3, file='al.txt', status='old')

read(3, \*)(al(i), i=1,al\_size)

close(3)

end if

end

subroutine input\_ia(ia)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real ia

dimension ia(dim+1)

open(4,file='ia.txt',status='old')

read(4,\*)(ia(i), i=1,dim+1)

close(4)

end

subroutine input\_vect(vect)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real vect

dimension vect(dim)

open(5, file='vector.txt', status='old')

read(5,\*)(vect(i), i=1,dim)

close(5)

end

subroutine output(res)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real res

dimension res(dim)

open(6, file='res.txt', status='old')

write(6,'(e11.4)')(res(i), i=1,dim)

close(6)

end

subroutine func(di,al,ia,vect,res)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i,j

real di,al,ia,vect,res

dimension di(dim),al(al\_size),ia(dim+1),

\*vect(dim),res(dim)

do i=1,dim

res(i)=res(i)+di(i)\*vect(i)

do j = int(ia(i)), int(ia(i+1))-1

res(i)=res(i)+al(j)\*vect(j+i-int(ia(i+1)))

res(j+i-int(ia(i+1)))=res(j+i-int(ia(i+1)))+al(j)\*vect(i)

end do

end do

end

1. **Тестирование**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Матрица | Вектор | Файлы | | | | | Результат | Назначение |
|  |  | al | ia | di | scal | vector |  | Матрица размерности 3 |
| 2 3 3 | 1 1 2 4 | 1 2 3 | 3 3 | 1 2 3 |
|  |  | 2 1 3 0 0 | 1 1 2 3 6 | 1 2 3 4 | 4 4 | 1 2 3 4 |  | Матрица размерности 4 |
|  |  | 25 1 30 0 0 | 1 1 2 3 6 | 0.1 0.2 0.3 4 | 4 7 | 1 0.02 3 0.04 |  | Матрица с вещественными элементами |
|  |  | 25 1 30 0 0 | 1 1 2 3 6 | 0.1 0.2 0.3 4 | 4 7 | 0 0 0 0 |  | Нулевой вектор |
|  |  |  | 1 1 1 1 1 | 0 0 0 0 | 4 0 | 1 2 3 4 |  | Нулевая матрица |
|  |  | … | … | … | 268435456  (238)  268435456  (238) | … |  | Недостаток памяти |

1. **Текст программы (матрица представлена в файлах прямого доступа)**

program main

implicit none

common/scal/dim,al\_size

common/lim/memlim

real mem

dimension mem(2\*\*28)

integer dim,al\_size

integer memlim

memlim = 2\*\*28

call input\_scal

call is\_mem\_enough

call input\_di(mem(1))

call input\_al(mem(dim+1))

call input\_ia(mem(dim+al\_size+1))

call input\_vect(mem(dim\*2+al\_size+2))

call func(mem(1),mem(dim+1),mem(dim+al\_size+1),

\*mem(dim\*2+al\_size+2),mem(dim\*3+al\_size+2))

call output(mem(dim\*3+al\_size+2))

end

subroutine input\_scal

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

open(1, file='scal.txt', status='old')

read(1,\*)dim,al\_size

close(1)

end

subroutine is\_mem\_enough

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

if ((dim\*4+1+al\_size).gt.memlim) then

print\*,'not enough memory'

pause

stop

end if

end

subroutine input\_di(di)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real di

dimension di(dim)

open(2, file='di.dat',status='old',

\*access='direct',recl=4,form='unformatted')

do i=1,dim

read(2,rec=i)di(i)

end do

close(2)

end

subroutine input\_al(al)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real al

dimension al(al\_size)

if (al\_size.ne.0) then

open(3, file='al.dat',status='old',

\*access='direct',recl=4,form='unformatted')

do i=1,al\_size

read(3, rec=i)al(i)

end do

close(3)

end if

end

subroutine input\_ia(ia)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real ia

dimension ia(dim+1)

open(4,file='ia.dat',status='old',

\*access='direct',recl=4,form='unformatted')

do i=1,dim+1

read(4,rec=i)ia(i)

end do

close(4)

end

subroutine input\_vect(vect)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real vect

dimension vect(dim)

open(5, file='vect.dat',status='old',

\*access='direct',recl=4,form='unformatted')

do i=1,dim

read(5,rec=i)vect(i)

end do

close(5)

end

subroutine output(res)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real res

dimension res(dim)

open(6, file='res.txt', status='old')

write(6,'(e11.4)')(res(i), i=1,dim)

close(6)

end

subroutine func(di,al,ia,vect,res)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i,j

real di,al,ia,vect,res

dimension di(dim),al(al\_size),ia(dim+1),

\*vect(dim),res(dim)

do i=1,dim

res(i)=res(i)+di(i)\*vect(i)

do j = int(ia(i)), int(ia(i+1))-1

res(i)=res(i)+al(j)\*vect(j+i-int(ia(i+1)))

res(j+i-int(ia(i+1)))=res(j+i-int(ia(i+1)))+al(j)\*vect(i)

end do

end do

end

1. **Генерация тестов большой размерности**

program main

implicit none

common/mat/dim

integer dim

read\*,dim

call make\_di

call make\_vect

call make\_al\_ia

call make\_scal

end

subroutine make\_di

implicit none

common/mat/dim

integer dim

integer i

open(1,file='di.dat',access='direct',recl=4,form='unformatted')

do i=1,dim

write(1,rec=i)i/1.

end do

close(1)

end

subroutine make\_vect

implicit none

common/mat/dim

integer dim

integer i

open(2,file='vect.dat',access='direct',recl=4,form='unformatted')

do i=1,dim

write(2,rec=i)i/1.

end do

close(2)

end

subroutine make\_al\_ia

implicit none

common/mat/dim

integer dim

integer i,j,k

open(3,file='al.dat',access='direct',recl=4,form='unformatted')

open(4,file='ia.dat',access='direct',recl=4,form='unformatted')

write(4,rec=1)1.

k=1

do i=2,dim

do j=1,i-1

write(3,rec=j+k-1)j/1.

end do

write(4,rec=i)k/1.

k=k+i-1

end do

write(4,rec=int(dim)+1)k/1.

close(4)

end

subroutine make\_scal

implicit none

common/mat/dim

integer dim

open(5,file='scal.txt')

write(5,\*)int(dim),int(dim\*(dim-1)/2)

close(5)

end

1. **Конвертация файлов прямого доступа в текстовые**

program main

implicit none

call output('vc')

end

subroutine output(file)

implicit none

character\*2 file

real in

integer err,i

i=1

open(1,file=file//'.dat',status='old',

\*access='direct',recl=4,form='unformatted',iostat=err)

if (err.eq.0) then

open(2,file=file//'.txt')

read(1,rec=i,iostat=err)in

do while(err.eq.0)

write(2,'(e11.4)')in

i=i+1

read(1,rec=i,iostat=err)in

end do

else

print\*,'such file doesnt exist'

end if

close(1)

close(2)

end

1. **Конвертация текстовых файлов в файлы прямого доступа**

program main

implicit none

call output('vc')

end

subroutine output(file)

implicit none

character\*2 file

real in

integer err,i

i=1

open(1,file=file//'.txt',status='old',iostat=err)

if (err.eq.0) then

open(2,file=file//'.dat',access='direct',recl=4,

\* form='unformatted',iostat=err)

read(1,\*,iostat=err)in

do while(err.eq.0)

write(2,rec=i)in

i=i+1

read(1,\*,iostat=err)in

end do

else

print\*,'such file doesnt exist'

end if

close(1)

close(2)

end

1. **Вывод матриц, представленных в профильном формате, в текстовый файл в плотном формате**

program main

implicit none

common/scal/dim,al\_size

common/lim/memlim

real mem

dimension mem(2\*\*28)

integer dim,al\_size

integer memlim

memlim = 2\*\*28

call input\_scal

call is\_mem\_enough

call input\_di(mem(1))

call input\_al(mem(dim+1))

call input\_ia(mem(dim+al\_size+1))

call output(mem(1),mem(dim+1),mem(dim+al\_size+1))

end

subroutine input\_scal

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

open(1, file='scal.txt', status='old')

read(1,\*)dim,al\_size

close(1)

end

subroutine is\_mem\_enough

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

if ((dim\*3+1+al\_size).gt.memlim) then

print\*,'not enough memory'

pause

stop

end if

end

subroutine input\_di(di)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real di

dimension di(dim)

open(2, file='di.dat',status='old',

\*access='direct',recl=4,form='unformatted')

do i=1,dim

read(2,rec=i)di(i)

end do

close(2)

end

subroutine input\_al(al)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real al

dimension al(al\_size)

if (al\_size.ne.0) then

open(3, file='al.dat',status='old',

\*access='direct',recl=4,form='unformatted')

do i=1,al\_size

read(3, rec=i)al(i)

end do

close(3)

end if

end

subroutine input\_ia(ia)

implicit none

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

integer i

real ia

dimension ia(dim+1)

open(4,file='ia.dat',status='old',

\*access='direct',recl=4,form='unformatted')

do i=1,dim+1

read(4,rec=i)ia(i)

end do

close(4)

end

subroutine output(di,al,ia)

common/scal/dim,al\_size

common/lim/memlim

integer dim,al\_size,memlim

real di,al,ia

dimension di(dim),al(al\_size),ia(dim+1)

integer i,j

open(1,file='res.txt')

do i=1,dim

do j=1,i-1-(ia(i+1)-ia(i))

write(1,'(3x,f4.2,$)')0.

end do

do j=ia(i),ia(i+1)-1

write(1,'(3x,f4.2,$)')al(j)

end do

write(1,'(3x,f4.2,$)')di(i)

do j=i+1,dim

if ((i-1-(ia(i+2)-ia(i+1))).ge.j) then

write(1,'(3x,f4.2,$)')0.

else

write(1,'(3x,f4.2,$)')al(int(ia(j))+i-1)

end if

end do

write(1,'(/,$)')

end do

close(1)

end