Speeding up R by using ISM-like calls

Junji NAKANO (The Institute of Statistical Mathematics, Japan)

and

Ei-ji NAKAMA (COM-ONE Ltd., Japan)

Outline

- What are ISM-like calls?
- Using ISM functions in R
- Benchmark examples
- System administration
- Concluding remarks

Two ISMs

- ISM: Intimate Shared Memory
  - is an optimization mechanism introduced first in Solaris 2.2
  - allows for the sharing of the translation tables involved in the virtual to physical address translation for shared memory pages
- ISM: the Institute of Statistical Mathematics
  - is a research organization for Statistics in Japan
  - has about 50 stuff members
  - owns supercomputer systems
    - SGI Altix3700 (Intel Itanium2, Red Hat Linux V.3)
    - HITACHI SR11000 (IBM Power4+, AIX 5L V5.2)
    - HP XC4000 (AMD Opteron, Red Hat Linux V.4)
  - uses R on these supercomputers
  - is a “real” center of Japanese R users. A “Virtual” center of them is RjpWiki (`http://www.okada.jp.org/RWiki/`)

ISM and TLB (1)

- All modern processors implement some form of a Translation Lookaside Buffer (TLB)
- This is (essentially) a hardware cache of address translation information
- Intimate Shared Memory (ISM) can make effective use of the hardware TLB in Solaris OS
  1. Enabling larger pages - 2-256MB instead of the default 4-8KB
  2. Locking pages in memory - no paging to disk
- Similar mechanisms are realized in many modern OSs
  - Linux - Huge TLB
  - AIX - Large Page
  - Windows - Large Page

ISM and TLB (2)

- The cost of translation between logical addresses and physical addresses is called “TLB miss” and sometimes becomes a bottle-neck
- These ISM-like calls may solve the problem
- We introduce the use of ISM-like mechanisms in R by adding a wrapper program on the memory allocation function of R and investigate the performance of them

First Benchmark

Following example is one of the most effective benchmarks of using the ISM-like function.

```
hilbert<-function(N){
  1/(matrix(1:N, N, N, byrow=T) + 0:(N - 1))
}
system.time(qr(hilbert(1000)),gcFirst=T)
ISM(T) # ISM enable
system.time(qr(hilbert(1000)),gcFirst=T)
```

OS / CPU	Without ISM	With ISM
Linux amd64 / Opteron 275	15.209	5.987
Linux amd64 / Xeon E5430	7.822	5.323

Using ISM (1)

Use function "ISM()".

ISM enable/disable

```
> ISM(on = TRUE,                        # enable ISM
+     minKB = ISM.status()$minKB,
+     maxKB = ISM.status()$maxKB)
>
> system.time(sort(1:1e8))              # a (meaningless)
>                                       # calculation example
>
> ISM(FALSE)                            # disable ISM
```

Using ISM (2)

Use an assignment operator ":=".

ISM assign

```
> ‘:=‘
function (x, value)
{
    onoff <- ISM.status()$status
    ISM(TRUE)
    on.exit(ISM(onoff))
    assign(deparse(substitute(x)), value,
           envir = parent.env(environment()))
}
<environment: namespace:base>
> foo <- matrix(rnorm(1024^2),1024,1024)
> system.time(foo.qr := qr(foo), gcFirst=T)
```

Checking ISM memory

Size of used memory is shown by “ISM.list()”.

ISM list

```
> ISM(T)
> system.time(sort(1:1e8))
> ISM.list()
    shmid        address      size
1 2949123 0x2aaaaac00000 400556032
2 2981892 0x2aaac2a00000 400556032
3 3014661 0x2aaada800000 400556032
> gc()
         used (Mb) gc trigger  (Mb)  max used   (Mb)
Ncells 157990  8.5     350000  18.7    350000   18.7
Vcells 204943  1.6  126367980 964.2 150219014 1146.1
> ISM.list()
NULL
```

Checking ISM Status

Status of ISM is shown by “ISM.status()”.

- support
  is TRUE if ISM is available in this environment
- status
  is TRUE if ISM is enabled
- minKB
  shows the minimum memory size for using ISM (Unit: KB)
- maxKB
  shows the maximum memory size for using ISM (Unit: KB)
- largepagesize
  shows the size of large page of the system (Unit: KB)

```
> ISM.status()
$support
[1] TRUE

$status
[1] TRUE

$minKB
[1] 1024

$maxKB
[1] 4194304

$largepagesize
[1] 2048
```

FFT and inverse FFT

In this example, ISM is not useful at all, probably because TLB miss seldom happens.

```
testfft<-function(n=1024){
    x<-as.complex(1:n)
    all.equal(fft(fft(x), inverse = TRUE)/ length(x), x)
}
system.time(testfft(1e7), gcFirst=T)
system.time(testfft(2^24),gcFirst=T)
```

OS / CPU	length	Without ISM	With ISM
Linux amd64 / Opteron 275	10^7	19.104	18.234
	2^{24}	39.119	47.023
Linux amd64 / Xeon E5430	10^7	13.080	12.154
	2^{24}	30.590	38.552

Least squares for large data

ISM is (very) useful in this example.

```
set.seed(123)
y<-matrix(rnorm(10000*5000),5000)
x<-matrix(runif(100*5000),5000)
system.time(fit<-lm(y~x),gcFirst=T)
```

OS / CPU	Without ISM	With ISM
Linux amd64 / Opteron 275	216.756	67.126
Linux amd64 / Xeon E5430	30.493	28.005

OS dependence

We execute 3 OSs on one machine. Results does not depend on OSs.

```
hilbert<-function(N){
  1/(matrix(1:N, N, N, byrow=T) + 0:(N - 1))
}
system.time(qr(hilbert(1e3)),gcFirst=T)
system.time(qr(hilbert(2ˆ10)),gcFirst=T)
```

OS / CPU	size	Without ISM	With ISM
Linux amd64 / Opteron 248	10^3	20.197	9.826
(gcc-4.1 -O2)	2^{10}	83.120	60.346
Solaris10 / Opteron 248	10^3	20.138	8.456
(Sun -xlibmil -xO5 -dalign)	2^{10}	71.194	57.181
Vista x64 / Opteron 248	10^3	22.74	10.12
(gcc-4.1 -O3)	2^{10}	78.08	53.81

CPU dependence

We execute one OS on 5 CPUs. Results depend on CPUs.

OS / CPU	size	Without ISM	With ISM
Linux-2.6.18 amd64 / Opteron 248	10^3	20.197	9.826
	2^{10}	83.120	60.346
Linux-2.6.18 amd64 / Opteron 275	10^3	15.209	5.987
	2^{10}	58.296	42.988
Linux-2.6.18 amd64 / Xeon E5430	10^3	7.822	5.323
	2^{10}	27.438	114.259
Linux-2.6.18 amd64 / Xeon 3040	10^3	12.555	8.983
	2^{10}	59.440	69.471
Linux-2.6.18 powerpc64 / Powerpc G5	10^3	27.214	26.220
	2^{10}	166.487	113.136

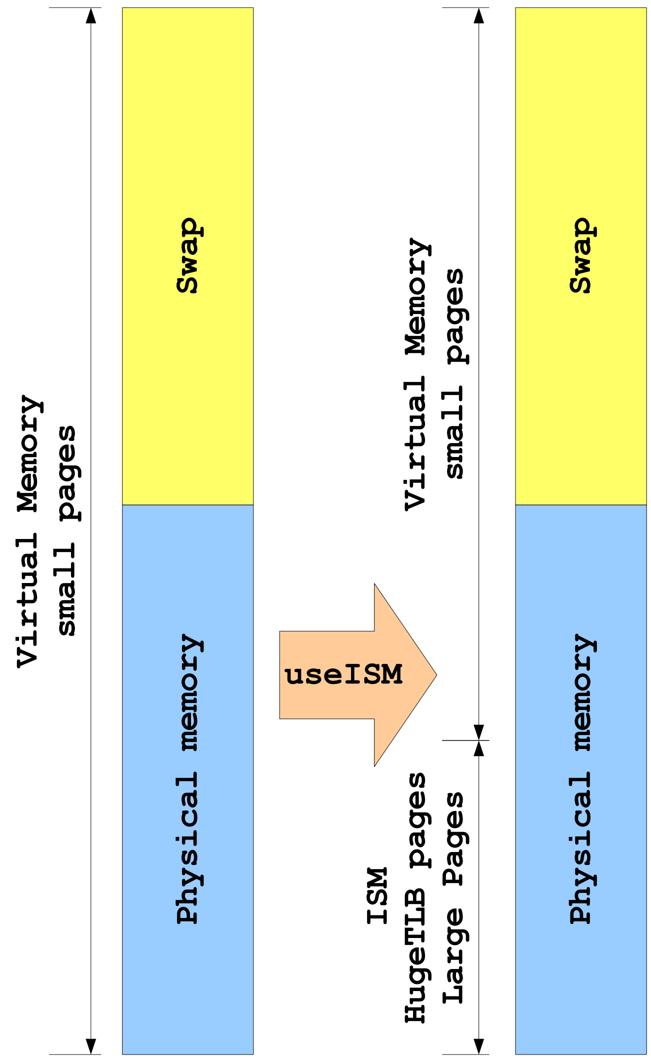
Install ISM to R

```
$ wget http://prs.ism.ac.jp/RISM/ism_2.7.1.patch
$ patch -p1 < ism_2.7.1.patch
```

By this patch, on

- UNIX,
  "–with-ism" is set to "yes" in configure
- Windows,
  "USE_ISM" is set to "yes" in src/gnuwin32/MKRules file

OS administration



ISM is not available by default except Solaris10.
To use ISM, We have to specify

- Resource management of users
- Memory size of HugeTLB pages

Note that HugeTLB pages generally are not used by usual programs.
Therefore, all physical memory may not be efficiently used.

OS administration - Solaris10

- Resource management of users and memory size for ISM are specified in “project” and reboot operation is required

```
projmod -K "project.max-shm-memory=
            (priv,2gb,deny)" group.staff
```

- Check status

```
$ /usr/bin/id -p
uid=500(ruser) gid=10(staff) projid=10(group.staff)
$ /usr/bin/prctl -n project.max-shm-memory
                 -i project group.staff
project: 10: group.staff
NAME    PRIVILEGE       VALUE    FLAG   ACTION
project.max-shm-memory
        privileged      2.00GB      -   deny
        system          16.0EB    max   deny
```

OS administration - Solaris8,9

- Resource management and memory size
  Edit /etc/system file, and reboot

```
set shmsys:shminfo_shmmax=2147483648
```

- Check status

```
$ /usr/sbin/sysdef |grep SHM
2147483648 max shared memory segment size (SHMMAX)
100 shared memory identifiers (SHMMNI)
```

OS Administration - Linux (1)

Setting of environments

- Debian Linux
  Set "Y" to [File systems]$\Rightarrow$ [Pseudo filesystems]$\Rightarrow$ [HugeTLB file system support] and rebuild the kernel
- Red Hat Linux
  The result of "ulimit -l" should be "unlimited"
  In /etc/security/limits.conf, add

```
*    -    memlock    unlimited
```

OS Administration - Linux (2)

- For Setting HugeTLB size, in /etc/sysctl.conf, add vm.nr_hugepages = 1024, and reboot
- Check status

```
$ cat /proc/meminfo |grep Huge
HugePages_Total:  1024
HugePages_Free:   1024
HugePages_Rsvd:      0
Hugepagesize:     2048 kB
```

OS Administration - Linux (3)

For setting SHM, edit /etc/sysctl.conf

- SHMMAX (Unit: byte)
  kernel.shmmax=2141198334
- SHMALL (Unit: page)
  kernel.shmall=522753

SHMALL is specified by the number of pages including both small pages and large pages. Thus, a large number can be used for it.

OS administration - AIX

(Not yet tested.)

- For setting HugeTLB size, set

```
# smitty tuning
lgpg_regions = 256
lgpg_size = 16777216
```

  and reboot.
- Check status

```
$ vmo -a | grep lgpg
lgpg_regions = 256
lgpg_size = 16777216
soft_min_lgpgs_vmpool = 0
```

In addition, several setting for SHM are required.

OS administration - Windows

- Resource management
  Start → Control Panel → Administrative Tools → Local Security Policy → Local Policy → User Rights Assignment
  In “Lock pages in memory”, add “administrator”
- For execution,
  “Run as administrator.” is required.

Windows Vista has no function to reserve LagePage. It usually runs many process. Therefore, we lack LargePage soon after booting. In some other OSs, LagePage is dynamically set. However, we also lack LargePage after long execution.

Concluding remarks

- Advantages
  - If “TLB miss” often happens, ISM is effective
  - If data are huge, ISM is effective.
- Disadvantages
  - Calculation time sometimes becomes large by using ISM
  - Memory usage sometimes becomes inefficient
- Other characteristics
  - Effects of ISM depend on CPU, not on OS
  - Precision and calculation order are not effected by ISM
  - Effects of ISM sometimes depend on values of data
  - If the compiler optimization is effectively used, ISM is not effective