Experiences with the Sparse Matrix-Vector Multiplication on a Many-core Processor

Juan C. Pichel

Centro de Investigación en Tecnoloxías da Información (CITIUS)
Universidade de Santiago de Compostela, Spain
21st Int. Heterogeneity in Computing Workshop

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

CITIUS

citius.usc.es

Outline

CITIUS

Outline

CITIUS

Single-chip Cloud Computer (SCC)

▷ The SCC is an experimental processor created by Intel Labs for many-core software research

▷ It consists of 48 independent x86 cores arranged in 24 tiles

Single-chip Cloud Computer (SCC)

Tile

▷ Two P54c Pentium cores

▷ Modified cache hierarchy: L1D and L1I: 16KB, L2: 256 KB

▷ **No coherency** among cores caches: software methods (flushing)

▷ **Message Passing Buffer** (MPB): 16KB (8KB per core), support message passing programming model

▷ Each tile has its own **frequency domain** (from 100 to 800 MHz)
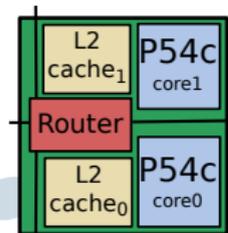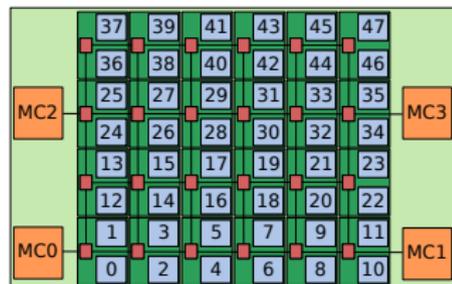
Single-chip Cloud Computer (SCC)

Mesh network

▷ Simple 2D grid that connects all tiles

▷ Data is routed first horizontally and then vertically through the network

▷ The network has its own **clock and power source** (800 MHz or 1.6 GHz)

▷ **Dynamic changes** during runtime are **not supported**
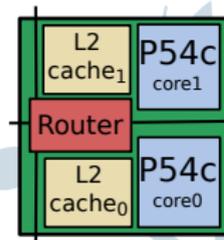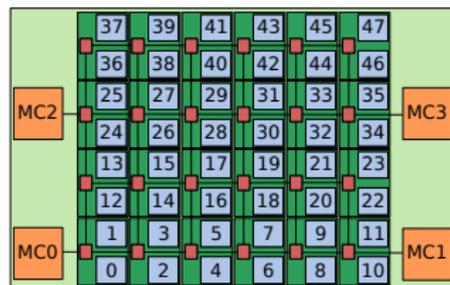
Single-chip Cloud Computer (SCC)

Main memory

▷ System admits 64 GB of main memory through 4 DDR3 memory controllers (MCs)

▷ Each core has its own private domain in this main memory (640 MB in our system)

▷ MCs attached to the routers of tiles at (0,0), (2,0), (0,5) and (2,5)

▷ Six tiles (12 cores) **share one MC** to access their private memory

▷ MCs operate on their own **clock and power source** (800 or 1066 MHz)
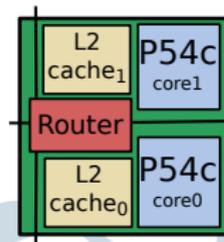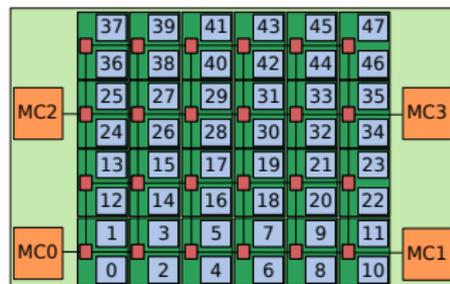
Single-chip Cloud Computer (SCC)

RCCE

▷ It is a simple message passing library

▷ Specifically designed to use the special architecture characteristics of the SCC (e.g. MPB)

▷ Two basic communication primitives: point-to-point and collective operations

▷ It also provides access to other entities (e.g. voltage controller)

▷ When executing a RCCE application, the cores to be used and their order can be configured differently

Sparse Matrix-Vector Multiplication (SpMV)


```
// SPMV using CSR format
for (i=0; i < n; i++) {
  reg = 0.0;
  for (j = ptr[i]; j < ptr[i+1]; j++)
    reg += da[j] * x[index[j]];

  y[i] = reg;
}
```

Row Pointers (ptr)

Column Indices (index)

Data values (da)
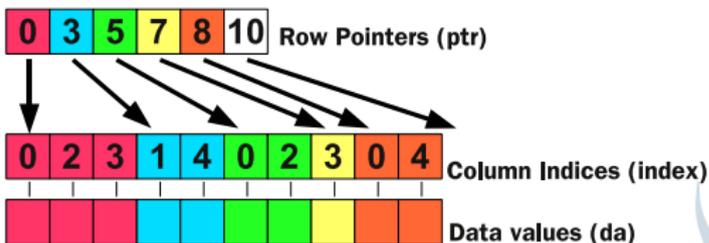
working set (in bytes) $= 4 \times ((n+1) + nnz) + 8 \times (nnz + 2 \times n)$

CITIUS

Outline

CITIUS

Experimental conditions

Matrices test set

▷ Thirty-two sparse matrices from different real problems that represent a variety of nonzero patterns

▷ $n \in [3140, 71505]$, $nnz \in [232633, 8767466]$, $nnz/n \in [7, 378]$ and $ws(MBytes) \in [2.9, 101.4]$

SCC platform

▷ It is based on Ubuntu Linux with a 2.6.32-24 kernel

▷ Cores, routers and memory clocked at the default speeds: 533 MHz, 800 MHz and 800 MHz respectively

▷ Codes were written in C and compiled with Intel's 8.1 Linux C compiler using RCCE 1.0.13

▷ Matrices are split row-wise with the same amount of nonzeros assigned to each UE

Mapping units of execution to cores

▷ Six tiles (12 cores) share MC, by default, partitioned in quadrants

▷ Memory latency of a core depends on the distance to the MC: $40C_{core} + 4 \times n_h \times 2C_{mesh} + 46C_{mem}$

▷ Only property to take into account: **number of mesh network hops** (n_h)
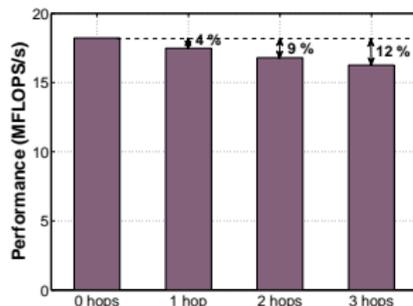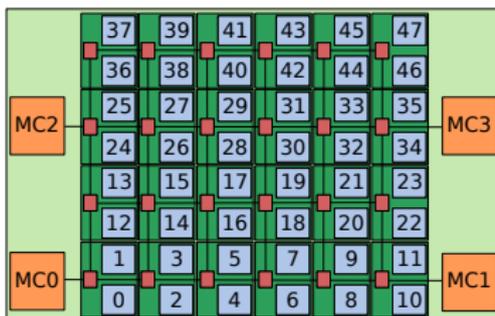
Mapping units of execution to cores
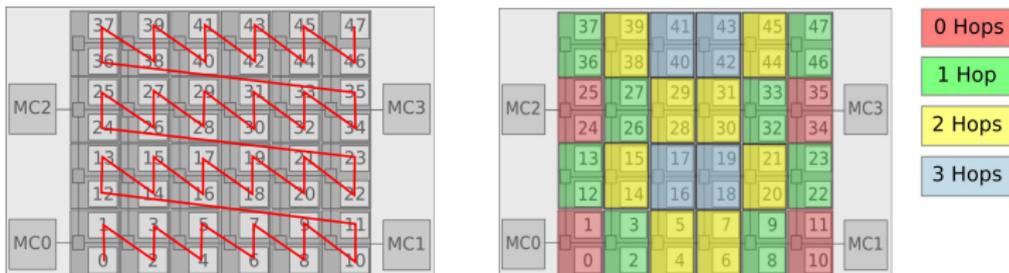
Different mappings of the UEs to cores: (a) standard and (b) considering the hops
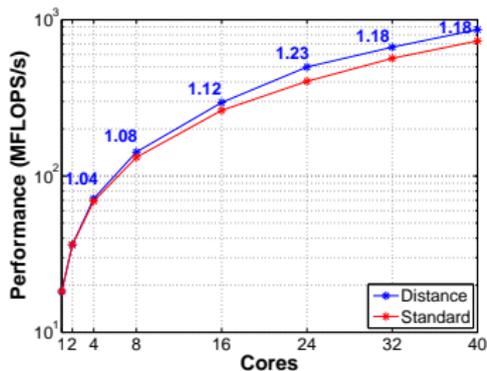
Influence of the working set size
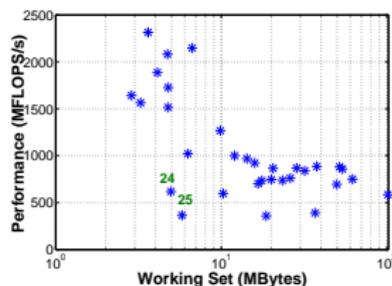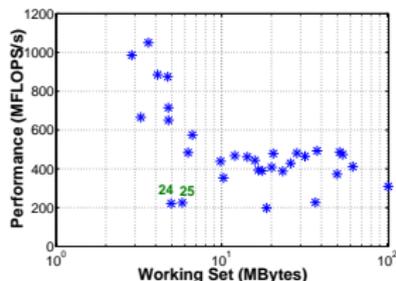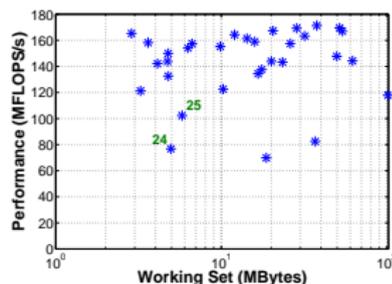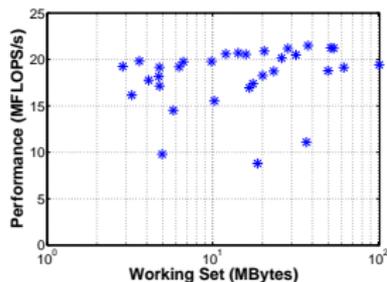
▷ Boost in performance with *cores* > 8: **matrices fit L2 cache**

▷ Matrices 24 and 25: very short row lengths (small *nnz*/*n*)

Influence of the irregular accesses

▷ SpMV performance: **low locality** caused by irregular accesses

▷ Modified version of the SpMV code: each reference to x access x[0]

▷ Performance differences between original and modified version caused by irregular accesses

▷ **Big impact on performance**: speedup over 10% in more than 50% of the matrices

▷ This behavior is not observed in other multicore systems: cache hierarchy of the SCC cores
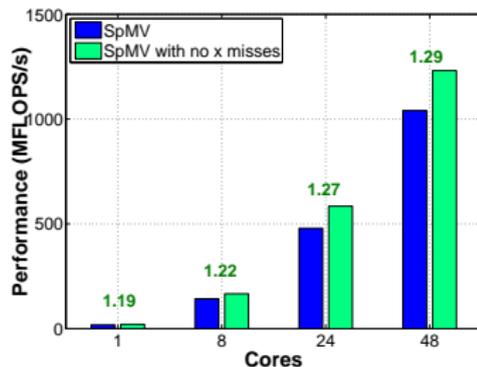
SCC configurations

▷ The SCC processor allows to change the cores, mesh network and memory clock frequency

▷ We consider three different configurations (frequencies in MHz):

- **$conf_0$** (default): 533, 800, 800
- **$conf_1$** : 800, 1600, 1066
- **$conf_2$** : 800, 1600, 800

SCC configurations
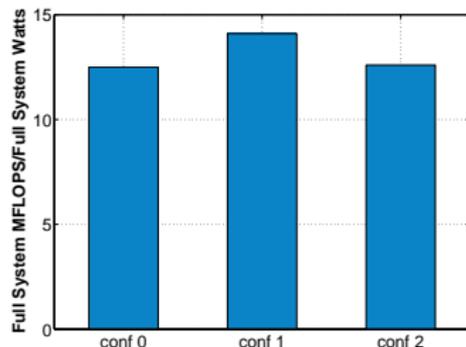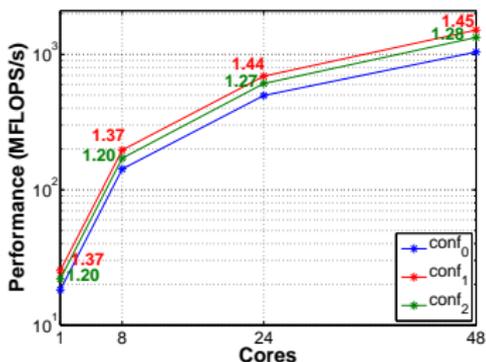
Comparison of the performance (a) and power efficiency (b)

▷ **conf$_1$** increases 30% the power consumption with respect to conf$_0$, but it is the **most efficient configuration**

▷ Efficiency of conf$_0$ and conf$_2$ is practically the same

Outline

CITIUS

Architectural Comparison

Evaluated systems

▷ **Intel Itanium2 Montvale**: This processor comprises two cores running at 1.6 GHz. The peak performance per core is 6.4 GFLOPS/s. Power consumption: 104 W (TDP).

▷ **Intel Xeon X5570**: It is a quad-core processor. Each core operates at 2.93 GHz, with 11.72 GFLOPS/s as peak performance per core. Power consumption: 95 W (TDP).

▷ **AMD Opteron 6174**: This processor consists of 12 cores running at 2.2 GHz. Power consumption: 80 W (ADP), 115 W (TDP).

▷ **NVIDIA Tesla C1060**: This GPU consists of 240 cores, with a double precision arithmetic peak performance of 78 GFLOPS/s. Power consumption: 187.8 W (TDP).

▷ **NVIDIA Tesla M2050**: It has 448 cores, with a double precision peak performance of 515.2 GFLOPS/s. Power consumption: 225 W (TDP).

Architectural Comparison
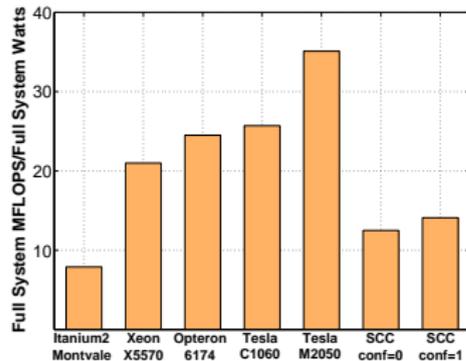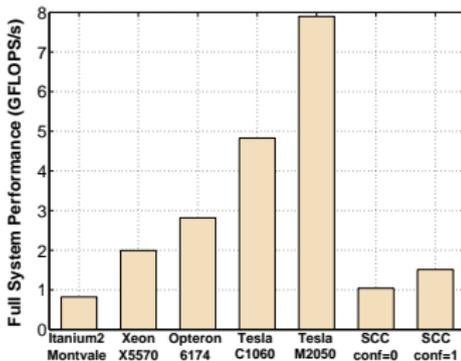
Comparison of the performance (a) and power efficiency (b)

▷ SCC outperforms Itanium2, behaves better in terms of power efficiency

▷ Xeon and Opteron efficiencies are similar to the observed with Tesla C1060

▷ **Best behavior overall**: Tesla M2050

Outline

CITIUS

Conclusions

A study of the behavior of the SpMV on the SCC many-core processor has been performed. Some of the most important observations are:

- ▷ Mapping the UEs to the cores with the lowest distance to memory increase the SpMV performance up to $1.23\times$
- ▷ A boost in the performance has been observed when the working set per core fits the L2 cache
- ▷ Unlike other multicore systems, the irregular accesses have a big impact on the SpMV performance
- ▷ Speedups up to $1.45\times$ were obtained using a different SCC configuration with respect to the default one
- ▷ SCC only outperforms the Intel Itanium2 in terms of performance and power efficiency

Thank you!!