Master Degree Program in Computer Science and Networking, 2014-15

High Performance Computing

3rd appello – June 16, 2015

Write your name, surname, student identification number (numero di matricola), e-mail. The answers can be written in English or in Italian. Please, present the work in a legible and readable form. All the answers must be properly and clearly explained.

1) Consider a client-server computation Σ with request-reply behavior and a constant number *n* of clients.

Show and explain the qualitative shape of the effective bandwidth $B_\Sigma(m)$ of Σ as a function of the parallelism degree *m* of the server, taking also into account the possible parallelism paradigm for the server.

Does an optimal value of *m* exist?

2) Consider a generic PE of a CMP-based shared memory architecture, with the following characteristics:

- directory-based cache coherence with basic invalidation semantics and inclusive two-level cache per PE. The cache coherence control is *entirely* delegated to the secondary cache units;
- no primitive support to indivisible sequences of memory accesses.

a) Describe the behavior of the PE interface unit W in terms of received and sent information and related actions.

b) Evaluate the maximum bandwidth and the internal latency of W.

3) A stream-based sequential module Q encapsulates an integer array A[M] with M = 3 Mega. Let x be the generic integer element of the input stream. The corresponding integer element y of the output stream is equal to the number of elements A[j] such that A[j] < x. Moreover, if A[j] < x, then A[j] is assigned to x.

The interarrival time is equal to $1.5M\tau$.

The architecture is specified below. The service time per instruction is equal to 2τ.

a) Determine the optimal parallelism degree of Q, assuming the worst case A[i] = x for every i.

b) Explain all the feasible parallel implementations, and determine the parallel implementation having the best bandwidth and latency. This issue must be studied carefully in terms of:
1. the impact of the memory hierarchy,
2. the possible modification of the parallelism degree with respect to the value determined in *a)*,
3. the qualitative impact of the parallel implementation on the under-load memory access latency.

For simplicity, assume that the interarrival time is independent from the parallel implementation.

Architectural specifications

- single-CMP SMP machine with 16 PEs and two MINFs;
- the internal interconnect is a *bidirectional ring* with topology: PE_0, …, PE_7, $MINF_0$, PE_8, …, PE_{15}, $MINF_1$;
- double-buffering communications;
- D-RISC CPU with 32K + 32K primary cache, 8-word blocks, and 512K inclusive secondary cache;
- directory-based cache coherence with home-flushing semantics;
- communication processor;
- 1-Teraword external memory with two interleaved macromodules, 8 modules each, clock cycle equal to 20τ, and link transmission latency equal to τ;
- LC run-time support at student choice, e.g. Rdy-Ack interprocess communication with I/O-based synchronization and exclusive-mapping.

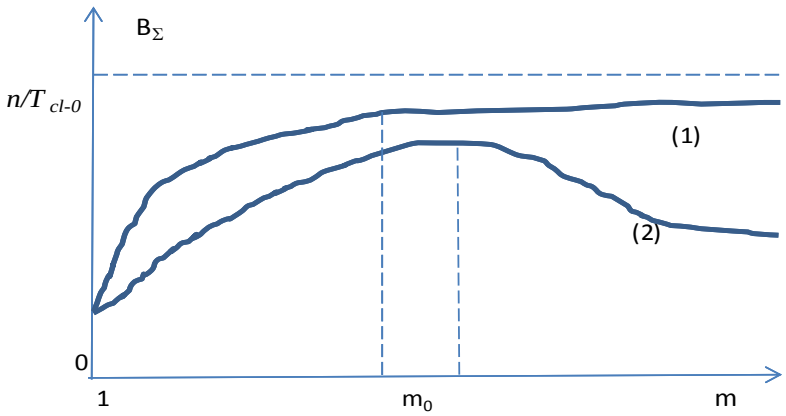
Solution

1) The effective bandwidth B_Σ of Σ is equal to the effective bandwidth of the set of n clients, which is equal to the effective bandwidth of the server, which is equal to the effective interarrival rate to the server (being $\rho < 1$ always):

$$B_\Sigma = \frac{n}{T_{cl}} = \frac{n}{T_{cl-0} + R_Q} = \frac{n}{T_{cl-0} + W_Q(\rho) + L_S}$$

The quantities affected by the parallelism degree m of the server are the server service time, thus the queue waiting time W_Q, and the server latency L_S.

For some parallelism paradigms, notably data parallel (and data flow), both service time and latency decrease by increasing m: in this case we have an asymptotic shape tending to n/T_{cl-0}: see figure, case 1.



For other parallel paradigms, notably farms and mainly pipeline, the latency increases with the parallelism degree. According to the relative weight of L_S compared to W_Q, either case 1 still occurs, or case 2 in which a maximum is even possible (see figure).

In practical situations, case 1 is the most likely. Of course, in case 1 an optimal value of m does not exist in theory: in practice, it is easy to recognize a value after which the bandwidth remains constant with known approximation degree.

2) W is connected, in input and in output, to C2, UC and to an internal interconnect interface unit SW. The behavior is described for each possible firmware message and consequent actions. According to the specifications, the processor synchronization doesn't require indivisible sequences of memory accesses (i.e., no *indiv* bit). In the following we assume 2-word physical addresses.

Messages from C2:

1. block reading request for a C2C access to another PE (home or owner) = (operation, physical address): transformed into a stream (header, 2 words for physical address), sent to SW one flit at the time. Latency: 1 clock cycle per flit;
2. block reading reply from this PE (home or owner) in response to a previous C2C reading request 1 = (operation, PE_dest_identifier, σ_1 block words one word at the time): transformed into a stream (header, σ_1 block words), sent to SW one flit at the time. Latency: 1 clock cycle per flit;
3. block writing request (write-through or write-back) to another PE (home - and possibly to owner if known) = (operation, physical address, one word, or σ_1 block words one word at the time): transformed into a stream (header, 2 words for physical address, one or σ_1 block words), sent to SW one flit at the time. Latency: 1 clock cycle per flit;
4. block writing reply from this PE (home or owner) in response to a previous synchronous writing request = (operation, physical address): transformed into a stream (header, 2 words for physical address), sent to SW one flit at the time. Latency: 1 clock cycle per flit;

5. invalidation request from this PE (home) to another PE = (operation, physical address): transformed into a stream (header, 2 words for physical address), sent to SW one flit at the time. Latency: 1 clock cycle per flit;
6. invalidation ack from this PE (home) in response to a previous invalidation request = (operation, physical address): transformed into a stream (header, 2 words for physical address), sent to SW one flit at the time. Latency: 1 clock cycle per flit;

Messages from UC:

7. interprocessor communication = (PE_dest_identifier, event, data_1, data_2): transformed into a stream (header, event, data_1, data_2), sent to SW one flit at the time. Latency: 1 clock cycle per flit.

Messages from SW:

8, 9 10, 11, 12, 13, 14 the same/complementary operations mentioned for C2 (1, 2, 3, 4, 5, 6) and for UC (7) can be invoked from other PEs in the form of streams, and sent, one word at the time, to C2 or to UC according to the firmware message type. Latency: 1 clock cycle per flit.

About the bandwidth: W has a nondeterministic behavior (3 input guards: from C2, UC and SW), which can be partially transformed into a parallel behavior:

- simultaneous requests from C2 and UC must be sequentialized into the unique SW interface;
- in *non-home* nodes a request from C2/UC and a request from SW can be executed in parallel, either in the case they start during the same clock cycle or during different clock cycles but overlapped.

This parallelism can be obtained by a single unit (parallelism in microinstructions) or, in order to minimize complexity, by two distinct units (one for output and one for input). Thus, the maximum bandwidth is 2 flits per clock cycle.

Notice that the latency of one clock cycle per flit is optimistic for requests from C2 and UC, because they might be in conflict. An accurate evaluation would require the knowledge of the relative probabilities: e.g., with the I/O-based Rdy-Ack run-time support, the probability is approximately the same.

3)

a) Let's start with the analysis of the sequential module. The ideal service time $T_{Q\text{-}id}$ is equal to the calculation time T_{calc}, since this last is $O(M)$ and the communication latency $L_{com}(1)$ is a (small) constant.

The D-RISC code of the calculation phase is:

```
        CLEAR   Ry
        CLEAR   Ri
        LOAD    Rvtg, 0, Rx
LOOP:   LOAD    RA, Ri, Ra
        IF ≥    Ra, Rx, CONT
        STORE   RA, Ri, Rx
        INCR    Ry
CONT:   INCR    Ri
        IF < Ri, RM, LOOP
        STORE   Rmsg, 0, Ry
```

The ideal service time in absence of cache faults is (assuming the worst case of A[i] = x for every *i*):

$$T_{Q-id-0} \sim 6\,M\,T_{instr} = 12\,M\,\tau$$

From the memory hierarchy point of view, array A is characterized by locality and by reuse from stream element to stream element. The working set is the whole array A. This reuse cannot be exploited for the sequential module, neither in C1 nor in C2. Thus, the cache fault penalty is:

$$T_{fault} = \frac{M}{\sigma_1} R_{Q0} = \frac{M}{\sigma_1} L_{extM-C1}(\sigma_1)$$

where $L_{extM-C1}(\sigma_1)$ is the C1-block transfer *base* latency from the external shared memory into C2 and, in parallel, into C1 (inclusive two-level cache). This latency is evaluated using the pipelined communication cost model, considering the CMP internal interconnect and the external memory macromodules directly connected to the CMP MINFs.

The average number of crossed links between a random-chosen PE and a MINF is equal to 2.5. The path of a block read request is C1, C2, W, internal net, MINF, IM, M, thus with distance d = 8.5. Therefore:

$$L_{extM-C1}(\sigma_1) = (s_{req} + s_{reply} + 2\,d - 4)T_{hop} + \tau_M = 70\,\tau$$

since s_{req}= 3, s_{reply}= 9, T_{hop} = 2τ (maximum hop along the path). In conclusion:

$$T_{fault} = 8.75\,M\,\tau$$

$$T_{Q-id} = T_{Q-id-0} + T_{fault} = 20.75\,M\,\tau$$

The optimal parallelism degree is:

$$n = \left\lceil \frac{T_{Q-id}}{T_A} \right\rceil = 14$$

The number of PEs is sufficient for 14 workers, and possibly for 1 or 2 service modules.

b) Q has a *map-reduce* structure with *internal state*, where map consists in the comparison and possible modification of A and reduce is applied to the addition of occurrences. This structure cannot be implemented as a farm. The following paradigms are feasible:

1. a *data-parallel with loop-unfolding pipeline*, consisting in 14 stages and no service module;
2. a *data-parallel with a centralized multicast* having service time $O(n) << T_A$ (no bottleneck), and 14 independent map workers, with the following possible implementations of reduce:
   2.1. *centralized* (one additional service module),
   2.2. *tree-structure mapped onto the workers set* (the reduce result is generated by the root-worker).

   In both cases, the *reduce* is executed *in pipeline* and *does not affect the bandwidth*, since any reduce implementation is not bottleneck.

All solutions have the same bandwidth, equal to the ideal one ($1/T_A$), with parallelism degree *n*. The best latency, $O(M/n)$ for the map phase, is achieved by the data-parallel solutions 2, while for the pipeline data-parallel the latency is $O(M)$. In particular, in solution 2.1 the centralized reduce implementation adds a term $O(n)$ to the latency due to the nondeterminism support in the centralized reduce, while in 2.2 the tree-structured reduce adds a term $O(\lg n)$.

From the under-load latency viewpoint, for all solutions (1, 2.1, 2.2) a *low-p* mapping implementation exists.

The data-parallel solutions 1 and 2 have an additional important feature: the ideal bandwidth can be achieved with a *lower degree of parallelism*. In fact, the partition size $g = M/n$ ~ 214K is such that *the reuse on A can be applied to the secondary cache* of each worker PE.

Now for each worker:

$$T_{w-0} = 12\,g\,\tau = 12\,\frac{M}{n}\,\tau$$

$$T_{w-fault} = \frac{g}{\sigma_1} L_{C2-C1}(\sigma_1) = \frac{\sigma_1 + 2}{\sigma_1}\,\frac{M}{n}\,\tau = 1.25\,\frac{M}{n}\,\tau$$

$$T_w = T_{w-0} + T_{w-fault} = 13.25\,\frac{M}{n}\,\tau$$

By imposing:

$$T_w = T_A$$

we obtain:

$$n_{data\ -par} = 9$$

Notice that a lower parallelism degree has a positive impact on power consumption too, i.e., on cost.

Moreover, the under-load parameter T_p is lower, because of the lower penalty for faults.

In conclusion, solution 2.2 can be considered the best for the global viewpoint of bandwidth, latency, and cost.