Correlating Workload Characterists to Performance Metrics for the Cray X-MP/Y-MP

By Walid Abu-Sufah and John L. Larson

TR 89-43

Correlating Workload Characteristics to Performance Metrics for the Cray X-MP/Y-MP

Walid Abu-Sufah
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia

John L. Larson
National Center for Supercomputing Applications
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign

This work was supported by
Center for Supercomputing Research and Development (CSRD),
and National Center for Supercomputing Applications (NCSA),
University of Illinois at Urbana-Champaign

Presented at the Fourth SIAM Conference on Parallel Processing for Scientific Computing, December 11-13, 1989

Abstract

Workload characterization is essential for performance evaluation studies. For multiprocessor supercomputers, this characterization usually consists of program measurements from uniprocessor execution (e.g, average vector length, percentage vectorization, etc.). There is no consistent, quantitative correlation between such characteristics and performance metrics across different programs. We present a methodology for defining and measuring characterization parameters for both single and multi-processor executions. Using several production codes, we show for the CRAY X-MP/Y-MP that these characterization parameters correlate consistently to observed performance metrics across different programs. Moreover, the correlation allows the identification of bottlenecks in system architecture that limit performance.

Key Words: supercomputers, multiprocessors, performance evaluation, workload characterization, performance metrics, Cray X-MP, Cray Y-MP

Performance Evaluation

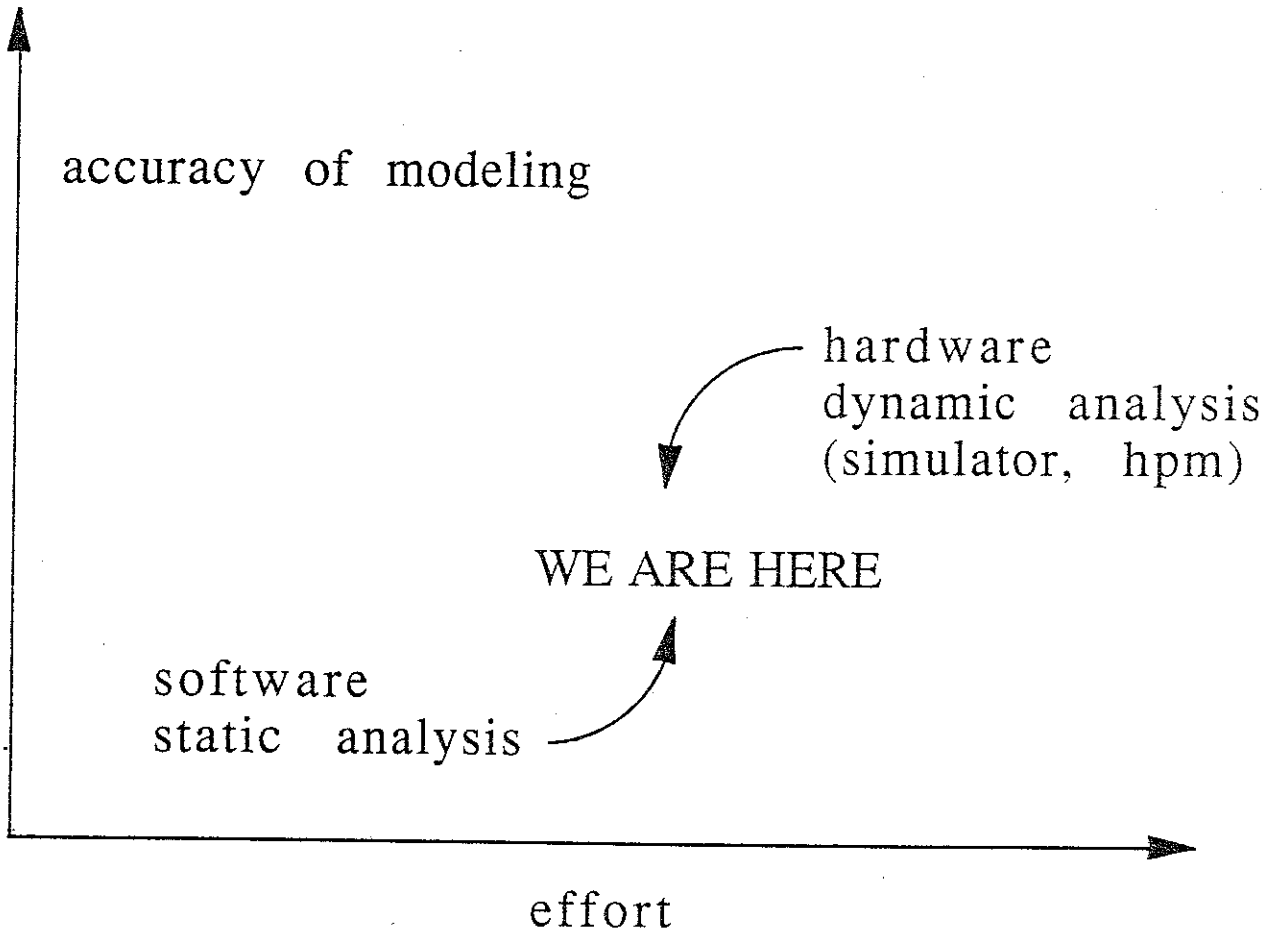
Axiom Number 1

A program that gets the right answer,

is **always** faster,

than one that doesn't.

What we're doing



Keywords

What is performance?

What is a program?

Performance definition

Rate of work performed by the (11) vector functional units. Overall work metric = MOPS

1. Computational functional units (8)

   a. Floating point units (3) - metric = MFLOPS

      floating point add
      floating point multiply
      reciprocal approximation

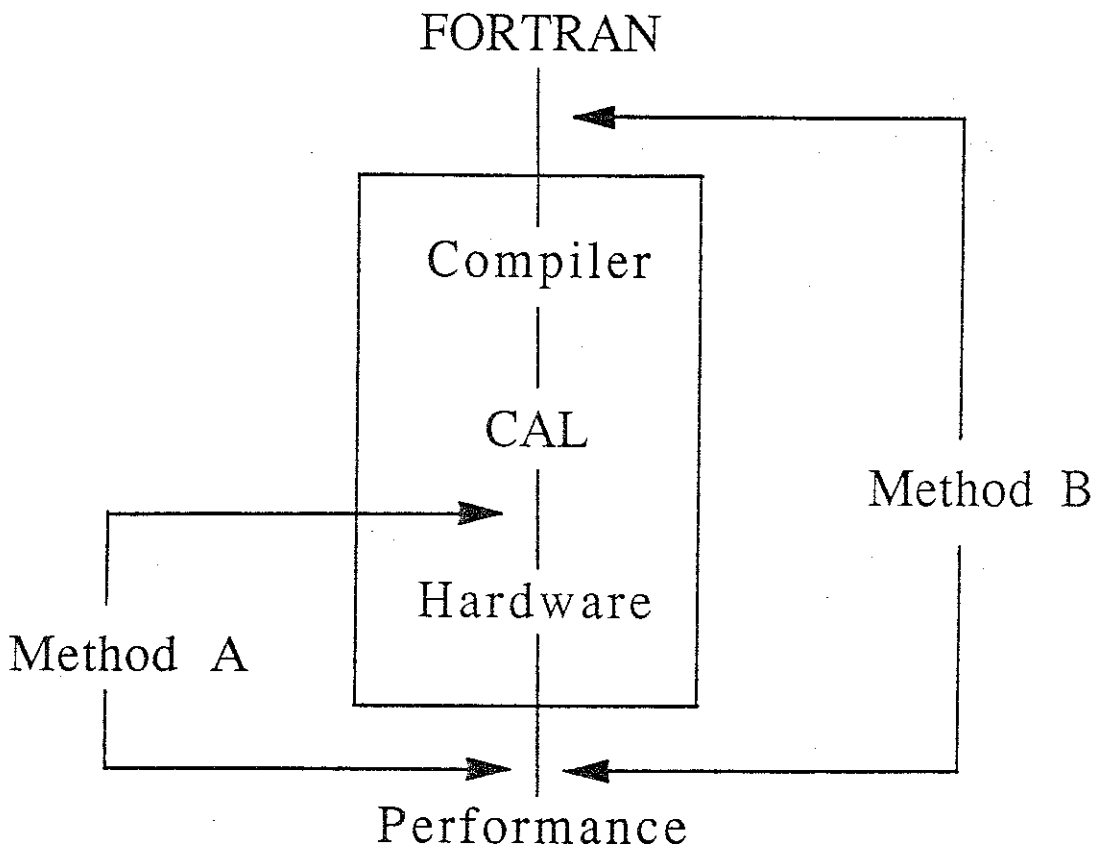
   b. Integer/logical units (5) - metric = MILOPS

      integer add
      logical (and, or, mask)
      shift
      population/parity
      second logical

2. Memory functional units (3) - metric = MMEMS

      load
      load (direct or indirect)
      store (direct or indirect)

Program definition



The assembly language code (CAL) is the program.

The source code is NOT the program, but only a display of opportunities.

The source code is NOT a true reflection of the work which will be performed.

WYSINWYG

What you see is NOT what you get

FORTRAN

```
      DO  10  I = 1,  50

10          IVEC( I ) = 8 * I
```

CAL

```
A7          50
VL          A7
S2          8
A0          CII$   @(0,1,2,...63)
V7          ,A0,1
A7          3
V6          V7 < A7    (I-1)*8
V5          S2 + V6    + 8
A0          @IVEC
,A0,1       V5
```

Since the X-MP has no vector integer multiply functional unit, something else has to be done.

Where to look for performance relationships

Program sets ⟷ Performance

0. Universe of all FORTRAN programs
1. An application area
2. Key algorithms
3. Kernals
4. Individual loops

infinite sets

---

finite sets

5. Chimes
6. Machine instructions
7. Electrons

Working with an infinite set is a big challenge.

Performance is an intrinsic attribute of an object, e.g. mass, electric charge, color.

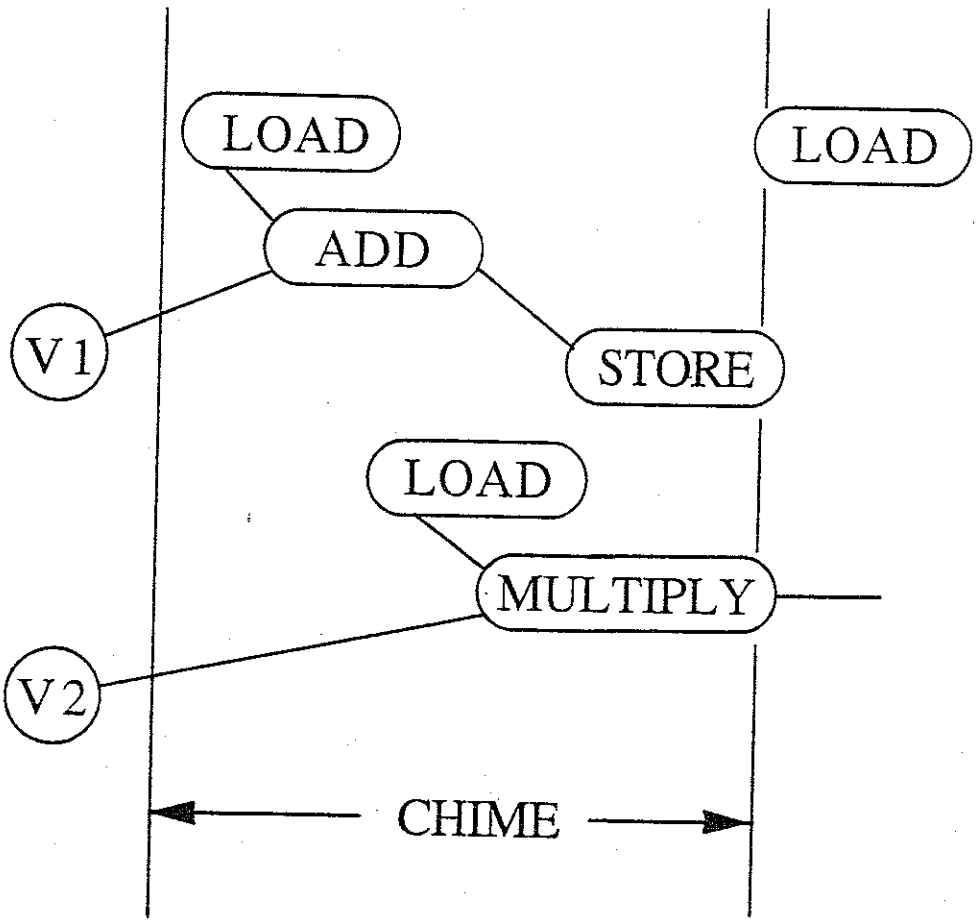
At what level is this attribute visible?

What is the essense of program performance?

Definitions

Chain - a sequence of related vector operations pipelined together.

Chime - a collection of one or more (partial) chains executing simultaneously.



▲ CHIME BREAKS ▲

Major chime types

low level: L L * + S (SAXPY)

high level: X Y Z = (5 2 3) (SAXPY)

X. vector operations

Y. computational operations

a. floating point
   1. add
   2. multiply
   3. reciprocal approximation

b. integer/logical
   1. integer add
   2. logical (and, or, mask)
   3. shift
   4. population/parity

Z. memory operations

a. load
b. load indirect
c. store
d. store indirect

Periodic table of major chime types

X Y Z = (vec-ops, comp-ops, mem-ops)

only 14 major types for MFLOPS metric

101	110		
202	211	220	
303	312	321	330
	413	422	431
		523	532

increasing flops per chime ⟶
(protons)

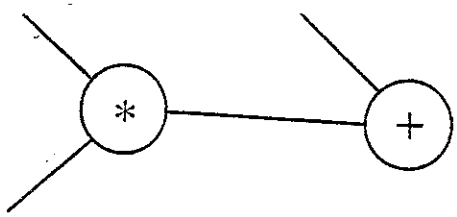
increasing memory operations per chime ↓
(neutrons)

Prospecting for performance

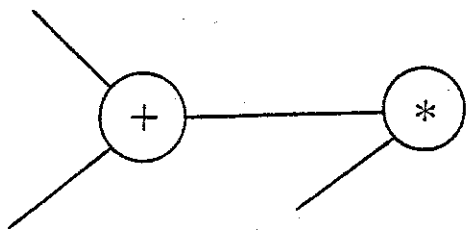
Pure Gold

(220)

(*, +) - isomer



(+, *) - isomer



Performance Model for Chimes

Common model for all pipeline machines

Perf (chime) =

F (vector length, peak speed, startup)

$$\text{MFLOPS} = r_{\infty} * \text{PIPE}\left(\text{vector length}, n_{1/2}\right)$$

Roger Hockney, Parallel Computers 2, 1988

Chime time

Cannot average performance of chimes over loop.

Must use harmonic average, weighted by time.

$$\text{Time (chime) } = \frac{\text{operations} * 10 ** 6}{\text{Mega-ops / second}}$$

Chemical Formulas

Pure chime loop (SAXPY - BLAS-1)

$$\frac{[\ (523)\ ]}{\text{ceil}\ (\ N/64\ )} \qquad \text{mem/flop} = 1.5$$

145 MFLOPS

Mixed chime loop (MXM - BLAS-3)

$$\frac{[\ (321)\ (220)\ ]}{\text{ceil}\ (\ N/64\ )} \qquad \text{mem/flop} = 0.25$$

225 MFLOPS

Decomposition of a loop into chimes

```
      DO 10  I = 1, 200

10  V( I ) = S + V( I )
```

chime type	chime		loop block
(413)	1	L + S L	1 VL=8
(312)	2	+ S L	2 VL=64
(312)	3	+ S L	3 VL=64
(211)	4	+ S	4 VL=64

original chime tool developed by Dick Hendrickson, Cray Research, 1983?

Application to a loop in a CFD code

for severe thunderstorm modeling

```
c     vertical advection terms for t,qv,qc,qr,km
CDIR$ IVDEP
      do 87004 k=1,nz1
      t1t (k,j)=-.5*rdz*
     1             (rrp(k)*w(k+1,j,i)*(t (k+1,j,i)-t ( k ,j,i))*fsw(k+1)
     2             +rrm(k)*w( k ,j,i)*(t ( k ,j,i)-t (k-1,j,i))*fsw( k ))
      qv1t(k,j)=-.5*rdz*fst(k)*
     1             (rrp(k)*w(k+1,j,i)*(qv(k+1,j,i)+qv( k ,j,i))
     2             -rrm(k)*w( k ,j,i)*(qv( k ,j,i)+qv(k-1,j,i)))
      qc1t(k,j)=-.5*rdz*fst(k)*
     1             (rrp(k)*w(k+1,j,i)*(qc(k+1,j,i)+qc( k ,j,i))
     2             -rrm(k)*w( k ,j,i)*(qc( k ,j,i)+qc(k-1,j,i)))
      qr1t(k,j)=-.5*rdz*fst(k)*
     1             (rrp(k)*w(k+1,j,i)*(qr(k+1,j,i)+qr( k ,j,i))
     2             -rrm(k)*w( k ,j,i)*(qr( k ,j,i)+qr(k-1,j,i)))
      km1t(k,j)=-.5*rdz*fst(k)*
     1             (rrp(k)*w(k+1,j,i)*(km(k+1,j,i)+km( k ,j,i))
     2             -rrm(k)*w( k ,j,i)*(km( k ,j,i)+km(k-1,j,i)))
      dvz(k)    =-2./3.*(edcm(k)*(km(k,j,i)+alowk*alkt(k)))**2
      duz(k)    =fsw(k+1)*((km(k,j,i)+km(k+1,j,i))+2.*alowk*alkw(k+1))
87004 continue
```

What function the loop performs
does not tell clearly about its performance.

What is this loop I see?

Chemical formula

```
36          V5          S5*RV6                    : : : : : :R:U: :
36          V1          V3+FV5                    : :R: :U: :U: : :

                      2 VECOPS
                      2  FLOPS
                      0  PORTS
                      0 LOGICS
                      0 INTEGS
                      0 INDIRS
36          V7          V2*RV1                    : :U:U: : : : :R:
36          ,A0,1       V7              DUZ       : : : : : : : :S:

                      2 VECOPS
                      1  FLOPS
                      1  PORTS
                      0 LOGICS
                      0 INTEGS
                      0 INDIRS
37          JAN         L7                        : : : : : : : : :

                     30 CHIMES
                    101 VECOPS
                     54  FLOPS
                     41  PORTS
                      0 LOGICS
                      6 INTEGS
                      0 INDIRS
                  3.367VECOPS/CHIME
                  1.800 FLOPS/CHIME
                  1.367 PORTS/CHIME
                  0.000 LOGIC/CHIME
                  0.200 INTEG/CHIME
                  0.000 INDIR/CHIME
```

(110) (211) (220) (321) (413) (422) (523)
1 2 5 6 2 5 3

\+ subtypes (421 010) (623 010) (311 010)
4 1 1

containing integer operations

Performance Prediction

Based on

1) current understanding of basic material properties of loop performance elements

2) enhanced model from one described here

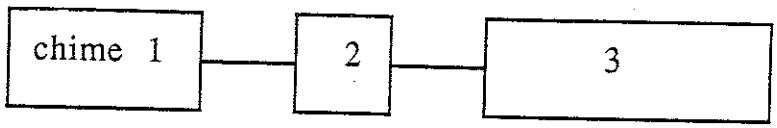
we have

Bob's loop:

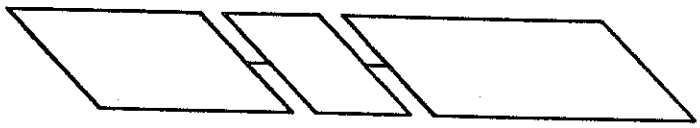
actual MFLOPS =	137.46	(hpm)
predicted MFLOPS =	88.93	(model)

Chemical bonding

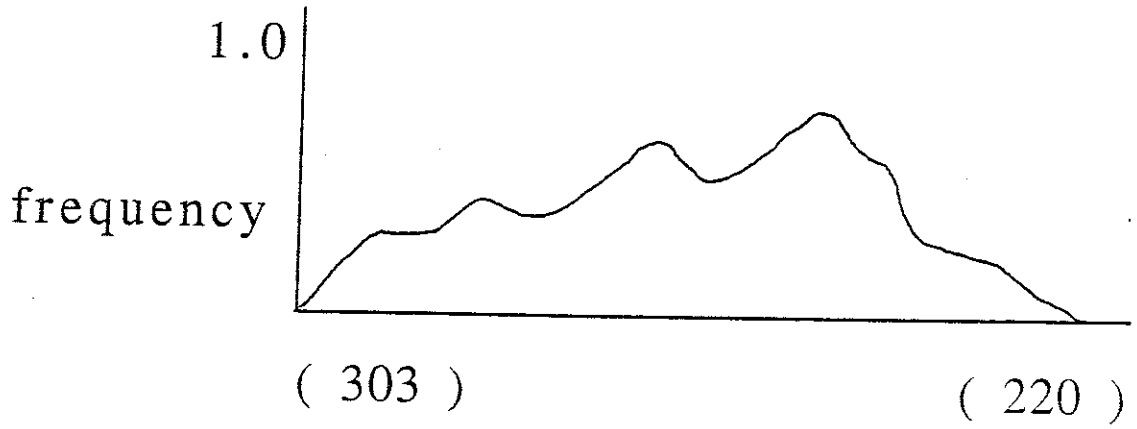
rectangle chime model



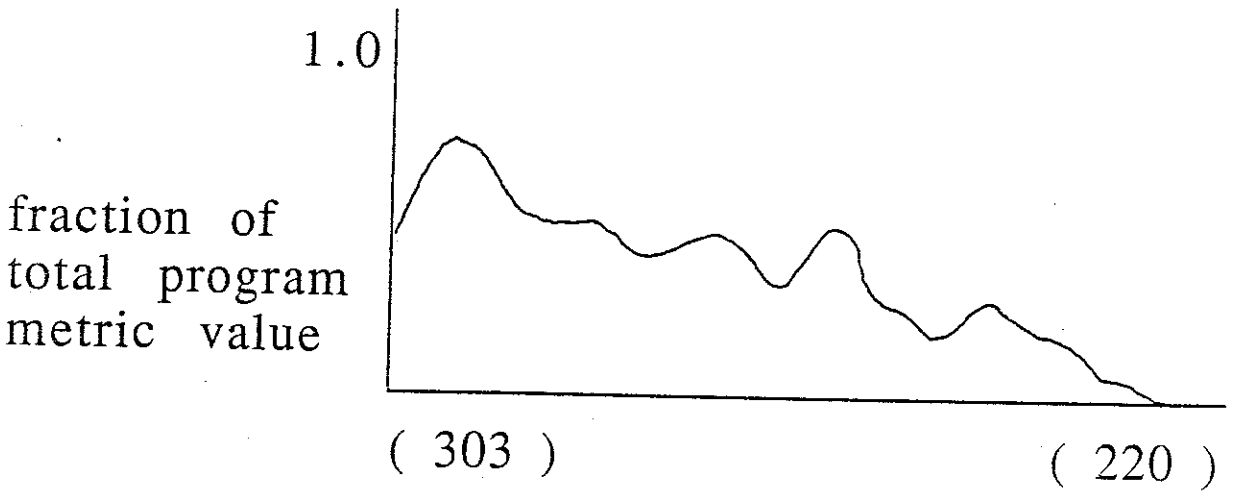
parallelogram chime model



Performance Profiling



chimes by increasing metric value



chimes by increasing metric value

(get the lead out!)

Future plans

* address shortcomings
* develop visualization interface for
  - vectorization training
  - understanding of optimizations
  - FORTRAN and CAL software development
  - identification of hardware/software bottlenecks

Performance Evaluation

Axiom Number 2

* It is reasonable to publish careful measurements, even if they cannot be understood or if the explanation seems far fetched.

* Not all such measurements should be published.

* It is difficult to know the difference.

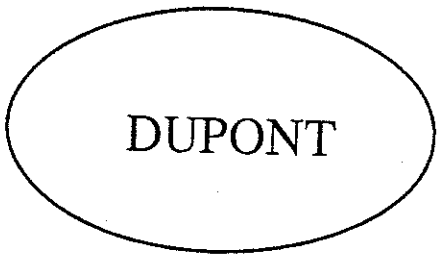
Margaret L. Simmons
Los Alamos National Laboratory

Conclusions

* After 8 years experience (6 1/2 at CRI), I have yet to understand vectorization.

* The methodology described here is applicable to all machines which use pipeline parallelism (of course, using architecture-specific details), and to all vector loops.

* investigators dealing with infinite sets will have a difficult time without encorporating knowledge of the basic elements of performance in the "programs" they use.



Better living

through chemistry