Appendix F

Source code of the C and Prolog benchmarks

%%

```
/* C version of tak benchmark */

#include <stdio.h>

int tak(x,y,z)
int x, y, z;
{
  int a1, a2, a3;
  if (x <= y) return z;
  a1 = tak(x-1,y,z);
  a2 = tak(y-1,z,x);
  a3 = tak(z-1,x,y);
  return tak(a1,a2,a3);
}

main()
{
  printf("%d\n", tak(24, 16, 8));
}
```

```
/* Prolog version of tak benchmark */

main :- tak(24,16,8,X), write(X), nl.

tak(X,Y,Z,A) :- X =< Y, Z = A.
tak(X,Y,Z,A) :- X > Y,
        X1 is X - 1, tak(X1,Y,Z,A1),
        Y1 is Y - 1, tak(Y1,Z,X,A2),
        Z1 is Z - 1, tak(Z1,X,Y,A3),
        tak(A1,A2,A3,A).
```

%%

```
/* C version of fib benchmark */

#include <stdio.h>

int fib(x)
int x;
{
  if (x <= 1) return 1;
  return (fib(x-1)+fib(x-2));
}
```

```
main()
{
  printf("%d\n", fib(30));
}

--------------------------------------------------------------------------------

/* Prolog version of fib benchmark */

main :- fib(30,N), write(N), nl.

fib(N,F) :- N =< 1, F = 1.
fib(N,F) :- N > 1,
        N1 is N - 1, fib(N1,F1),
        N2 is N - 2, fib(N2,F2),
        F is F1 + F2.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/* C version of hanoi benchmark */

#include <stdio.h>

han(n,a,b,c)
{
   int n1;

   if (n<=0) return;
   n1 = n-1;
   han(n1,a,c,b);
   han(n1,c,b,a);
}

main()
{
  han(20,1,2,3);
}

--------------------------------------------------------------------------------

/* Prolog version of hanoi benchmark */

main :- han(20,1,2,3).

han(N,_,_,_) :- N=<0.
han(N,A,B,C) :- N>0,
        N1 is N - 1,
        han(N1,A,C,B),
        han(N1,C,B,A).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/* C version of quicksort benchmark */
```

```
#include <stdio.h>

int ilist[50] = {27,74,17,33,94,18,46,83,65, 2,
                 32,53,28,85,99,47,28,82, 6,11,
                 55,29,39,81,90,37,10, 0,66,51,
                  7,21,85,27,31,63,75, 4,95,99,
                 11,28,61,74,18,92,40,53,59, 8};

int list[50];

qsort(l, r)
int l, r;
{
   int v, t, i, j;

   if (l<r) {
        v=list[l]; i=l; j=r+1;
        do {
            do i++; while (list[i]<v);
            do j--; while (list[j]>v);
            t=list[j]; list[j]=list[i]; list[i]=t;
        } while (j>i);
        list[i]=list[j]; list[j]=list[l]; list[l]=t;
        qsort(l,j-1);
        qsort(j+1,r);
   }
}

main()
{
   int i, j;

   for(j=0; j<10000; j++) {
       for(i=0;i<50;i++) list[i]=ilist[i];
       qsort(0,49);
   }
   for(i=0; i<50; i++) printf("%d ",list[i]);
   printf("\n");
}
```

```
/* Prolog version of quicksort benchmark */

main :- range(1,I,9999), qsort(_), fail.
main :- qsort(S), write(S), nl.

range(L,L,H).
range(L,I,H) :- L<H, L1 is L+1, range(L1,I,H).

qsort(S) :- qsort([27,74,17,33,94,18,46,83,65, 2,
                   32,53,28,85,99,47,28,82, 6,11,
                   55,29,39,81,90,37,10, 0,66,51,
                    7,21,85,27,31,63,75, 4,95,99,
```

```
                   11,28,61,74,18,92,40,53,59, 8],S,[]).

qsort([X|L],R,R0) :-
        partition(L,X,L1,L2),
        qsort(L2,R1,R0),
        qsort(L1,R,[X|R1]).
qsort([],R,R).

partition([Y|L],X,[Y|L1],L2) :- Y=<X, partition(L,X,L1,L2).
partition([Y|L],X,L1,[Y|L2]) :- Y>X,  partition(L,X,L1,L2).
partition([],_,[],[]).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```