Technical University of Cluj-Napoca
Faculty of Automation and Computer Science

Parallel Programming

Laboratory 6

~ 2022 ~

Balázs Benedek

Group 30444/1

Problem 1:

Value = 8500

M = 6 * 1.3 = 7.8

PROCESSES	Execution time [milliseconds]	[Relative] Speedup S(n) = T(1)/T(n)	[Relative] Efficiency E(n) = S(n) / M
precision: 1000			
1	3574	1	0.1282051282
6	1008	3.545634921	0.4545685796
12	784	4.558673469	0.5844453166
24	775	4.611612903	0.5912324235
precision: 2000			
1	38129	1	0.1282051282
6	11197	3.40528713	0.4365752731
12	9483	4.020774017	0.5154838483
24	8788	4.338757396	0.5562509483
precision: 3000			
1	177589	1	0.1282051282
6	43202	4.110666173	0.5270084837
12	37022	4.796850521	0.6149808361
24	35540	4.996876759	0.6406252255
precision: 664			
1	784	1	0.1282051282
6	219	3.579908676	0.4589626507
12	157	4.993630573	0.6402090479
24	148	5.297297297	0.6791406791

TABLE 1. Performance parameters for Problem 1

f(1000):
43466557686937456435688527675040625802564660517371780402481729089536555417949051890403879840079255169295922593080322634775209689623239873322471161642996440906533187938298969649928516003704476137795166849228875

f(2000):
42246963333923048787067256023414827825798528402506810980102801373143085843701307072412359963914151108844608753890960360764019471164359602927198331259873732625355580260699158591522949245390499872225679531698287448247299226390183371677806060701161549788671987985831146887087626459736908672288402365442229524334796448013951534956297208765265606952980649984197744872015561280266540455417171788193032402520431208251681712 5

f(3000):
41061588630797126033356837871926710522012510863736925240888543092690558427411340373133049166085004456083003683570694227458856936214547650267437304544685216048660629249736050346977345373319688740584725529008204908690751262205905454219588975803110922267084927479385953913331837124479554314761107327624006673793408519173181099320170677683893476676477873950217447026862782091855384222585830640830166186290035826685723821023580250435195147299791967652400478423637645334726836415264834624584057321424141993791724291860263981009786694239201540462015381867142573983507485139642113998271364067958117845819865869228596804324365670979600 0

Code:

```
:- dynamic memfib/2.

worker(TID, NbThreads, IStart, IStop) :-
    Start is IStart + TID,
    fibonacci(Start, IStop, NbThreads).

createThreads(N) :-
    createThreads(0, N).

createThreads(N, N) :- !.
createThreads(I, N) :-
    I < N,
    NewI is I + 1,
    thread_create(worker(I, N, 0, 664), IDX, []),
    createThreads(NewI, N),
    thread_join(IDX, _).
```

```
fibonacci(I, N, NbThreads) :-
    I =< N, !,
    % thread_self(Id),
    PREC is N // 2,
    sqrt5(PREC, SQRT5),
    % phi (1 + sqrt(5)) / 2
    PHI_NOM is 10 ^ PREC + SQRT5,
    PHI_DENOM is 10 ^ PREC * 2,
    % PHI / sqrt5 <=> PHI_NOM * sqrt5, PHI_DENOM * 5

    PHI_NOM_POW is PHI_NOM ^ I * SQRT5,
    PHI_DENOM_POW is PHI_DENOM ^ I * 5 * 10 ^ PREC,
    PHI_NOM_FINAL is 2 * PHI_NOM_POW + PHI_DENOM_POW,
    PHI_DENOM_FINAL is 2 * PHI_DENOM_POW,
    RES is PHI_NOM_FINAL // PHI_DENOM_FINAL,

    assertz(memfib(I, RES)),

    II is I + NbThreads,
    fibonacci(II, N, NbThreads).

% hax
sqrt5(N, R) :-
    A is 5 * 10 ^ (2 * N),
    nth_integer_root_and_remainder(2, A, R, _).

% calculate elapsed time using the get_time for executing the fibonacci with N
threads // main function
threadedFibonacci(N, T) :-
    get_time(T0),
    createThreads(N),
    get_time(T1),
    T is T1 - T0.
```