Parallel Congruence Closure SAT solver

Enrico Martini, VR445204

***Abstract*—In this report is presented a parallel implementation of a congruence closure algorithm for deduction in ground equational theories, able to solve a set of clauses in the quantifiers free fragment of first order logic, based on equality among variables, constants, function applications, recursive data structures with their elements and elements of arrays.**

I. Introduction

The first theory considered is the class of SMT problems is called EUF (Equality with Uninterpreted Functions), containing atoms that are equalities between terms built over uninterpreted function symbols. EUF (i.e., SAT modulo the theory of congruences) is important in applications such as the verification of pipelined processors, where, if the control is verified, the concrete data operations can be abstracted by uninterpreted function symbols. [1] It is the most important theory because its congruence closure algorithm is the core of the entire solver. The implemented algorithm also integrates the theory of lists $\mathcal{T}_{cons}$.

II. Methodology

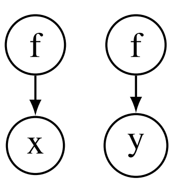
A. Algorithm

The most interesting feature of this implementation is the organization of the information within the data structures, shaped to be intuitive and efficient.

```
class Node{
private:
    std::string fn;
    uint_fast16_t id;
    std::vector<uint_fast16_t> args;
    uint_fast16_t find;
    std::vector<uint_fast16_t> ccpar;
};
```

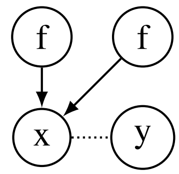
B. Equality theory congruence closure example

```
x=y&f(x)!=f(y)
```



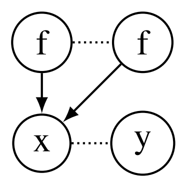
node	find	ccpar
0x	0	2
1y	1	3
2f->0	2	-
3f->1	3	-

```
MERGE 0 1
UNION 0 1
MERGE 2 3 ?
CONGRUENT 2 3 = 1
```

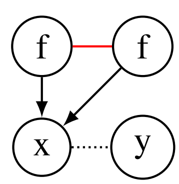


node	find	ccpar
0x	0	23
1y	0	-
2f->0	2	-
3f->1	3	-

```
MERGE 2 3
UNION 2 3
```



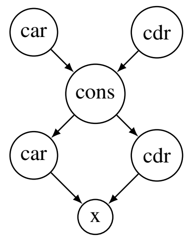
node	find	ccpar
0x	0	23
1y	0	-
2f->0	2	-
3f->1	2	-



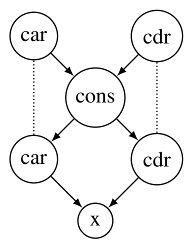

```
UNSAT
```

C. List theory congruence closure example

```
atom(x)&cons(car(x),cdr(x))=x
```

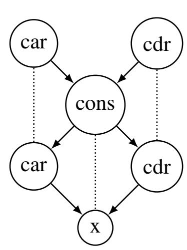



```
MERGE 4 1
UNION 4 1
MERGE 5 2
UNION 5 2
```

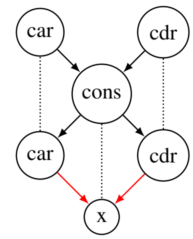


node	find	ccpar
0x	0	12
1car->0	4	–
2cdr->0	5	–
3cons->12	3	45
4car->3	4	3
5cdr->3	5	3

```
MERGE 3 0
UNION 3 0
MERGE 4 1 ?
MERGE 4 2 ?
CONGRUENT 4 2 = 0
MERGE 5 1 ?
CONGRUENT 5 1 = 0
MERGE 5 2 ?
```



node	find	ccpar
0x	3	–
1car->0	4	–
2cdr->0	5	–
3cons->12	3	4512
4car->3	4	3
5cdr->3	5	3



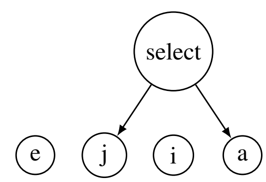

```
Euality theory passed
UNSAT
```

D. Array theory congruence closure example

```
e=select(store(a,i,e),j)&select(a,j)!=e
```

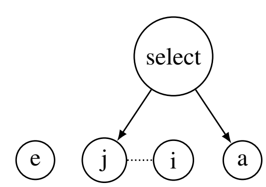
```
detected store
```

```
1: e=e&j=i&select(a,j)!=e
2: e=select(a,j)&j!=i&select(a,j)!=e
```

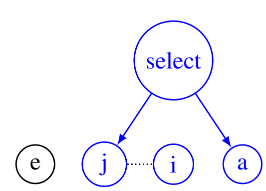


node	find	ccpar
0e	0	–
1j	1	4
2i	2	–
3a	3	4
4select->31	4	–

```
MERGE 0 0
MERGE 1 2
UNION 1 2
```



node	find	ccpar
0e	0	–
1j	1	4
2i	1	–
3a	3	4
4select->31	4	–



```
Euality theory passed
SAT
```

III. Validation

IV. Benchmarks

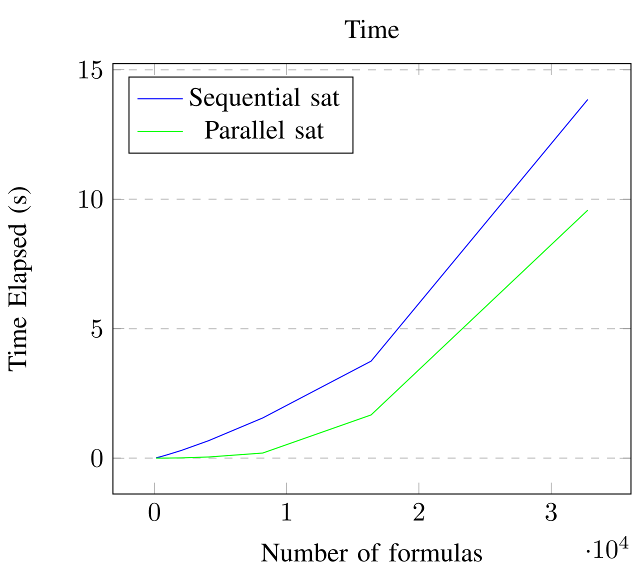
TABLE I
PERFORMANCE RESULTS WITH SIMPLE ALGORITHM.

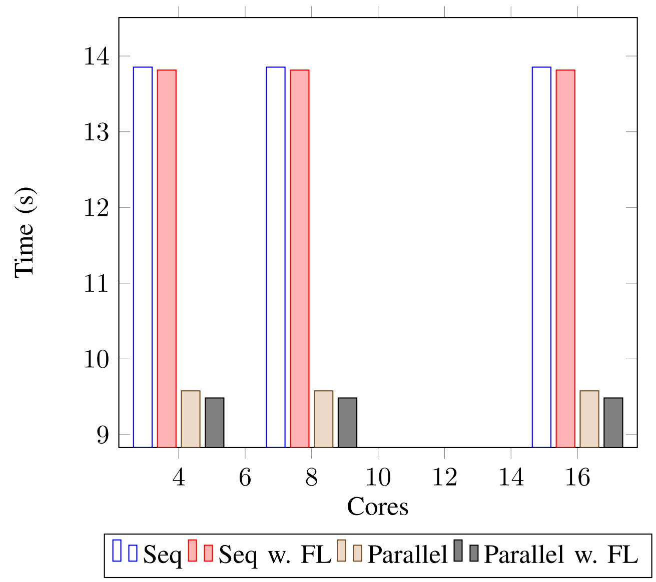
Test	#Formulas	Sequential (s)	Parallel (s)	Speedup
7	128	0,0117	0,0001	82,1
8	256	0,0290	0,0003	100,1
9	512	0,0612	0,0008	78,8
10	1024	0,1374	0,0026	52,1
11	2048	0,2988	0,0103	29,1
12	4096	0,6736	0,0442	15,3
13	8192	1,5526	0,1968	7,9
14	16384	3,7465	1,6690	2,2
15	32768	13,8530	9,5786	1,4

TABLE II
PERFORMANCE RESULTS WITH USE OF FORBIDDEN LIST

Test	#Formulas	Sequential (s)	Parallel (s)	Speedup
7	128	0,0128	0,0001	97,3
8	256	0,0273	0,0003	107,6
9	512	0,0605	0,0007	88,6
10	1024	0,1341	0,0024	57,0
11	2048	0,2964	0,0099	29,9
12	4096	0,6639	0,0430	15,5
13	8192	1,5309	0,1854	8,3
14	16384	3,7426	1,7264	2,2
15	32768	13,8145	9,4844	1,5

V. Performance Analysis





VI. Conclusion

References

[1] R. Nieuwenhuis and A. Oliveras, "Fast congruence closure and extensions," *Information and Computation*, vol. 205, no. 4, pp. 557 – 580, 2007. Special Issue: 16th International Conference on Rewriting Techniques and Applications.