Synthesis of Strategies for Impartial Two-Person Games

Alberto Pettorossi (Univ. Tor Vergata, Rome, Italy)

Maurizio Proietti (INSI-CNR, Rome, Italy)

IFIP WG 2.1 – Namur, Melgium

December 11th-15th, 2006

Initial Program

```
win(0,M).                 % wins who finds the table empty
win(s(N),1)     :-        ¬ win(N,1), ¬ win(N,2).
win(s(s(N)),2) :-         ¬ win(N,1), ¬ win(N,2).
```

Initial Program with Types

1. win(0,M). % wins who finds the table empty
2. win(s(N),1) :- nat(N), ¬ win(N,1), ¬ win(N,2).
3. win(s(s(N)),2) :- nat(N), ¬ win(N,1), ¬ win(N,2).

4. nat(0).
5. nat(s(N)) :- nat(N).

6. move(1).
7. move(2).

By definition, unfolding, folding steps we get:

Definite, Nondeterministic Final Program

```
win(0,M).                   % definite program:
win(s(N),1) :- new1(N).     %    no negation in the bodies
win(s(N),2) :- new2(N).     % nondeterministic program

new1(s(N)) :- new3(N).      % nondeterministic program
new1(s(N)) :- new4(N).      % nondeterministic program
new2(s(N)) :- new1(N).
new3(0).
new4(0).
new4(s(N)) :- new5(N).
new5(s(N)) :- new1(N).
```

The Derivation ...

- initial definition -

w(N,M) :- win(N,M).

- unfold -

w(0,M).

w(s(N),1) :- nat(N), ¬ win(N,1), ¬ win(N,2).

w(s(s(N)),2) :- nat(N), ¬ win(N,1), ¬ win(A,2).

- define -

new1(N) :- nat(N), ¬ w(N,1), ¬ w(N,2).

new2(s(N)) :- nat(N), ¬ w(N,1), ¬ w(N,2).

- fold -

w(0,M).

w(s(N),1) :- new1(N).

w(s(N),2) :- new2(N).

...

After the Determinization Strategy we get:

```
det_win(0,M).                          % definite program
det_win(s(N),M) :- new2(N,M).          % deterministic program

new2(s(N),M) :- new3(N,M).

new3(0,1).
new3(s(N),M) :- new4(N,M).

new4(0,2).
new4(s(N),M) :- new5(N,M).

new5(s(N),M) :- new3(N,M).
```

The idea of Determinization

```
a :- b
a :- c
```

% a is nondeterministic

becomes

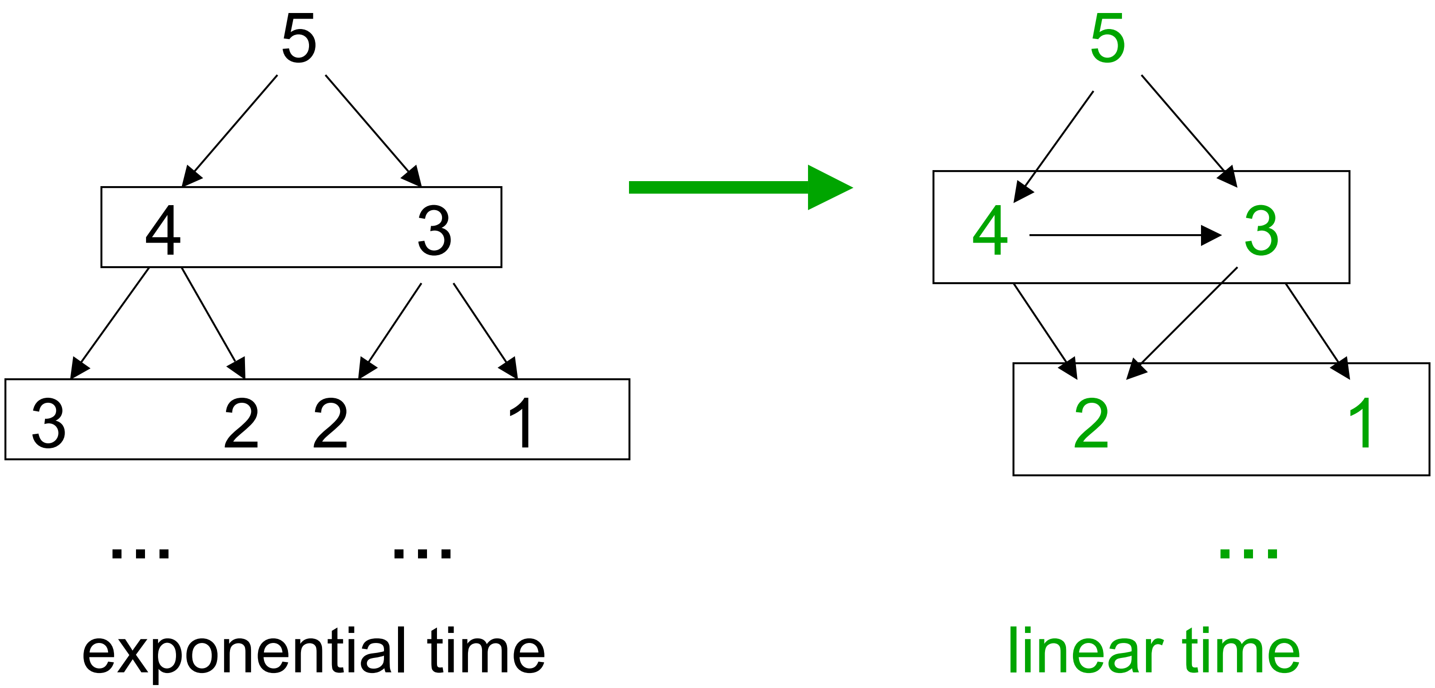
```
a :- new
new :- b
new :- c
```

% a is deterministic

If we can fold new then we avoid nondeterminism.

Determinization: *from exponential to linear*

(as in Fibonacci)



Further Improvements (1)

Actually, ... new2 is equal to new5. Eliminating new5:

```
det_win(0,M).
det_win(s(N),M) :- new2(N,M).

new2(s(N),M) :- new3(N,M).
new3(0,1).
new3(s(N),M) :- new4(N,M).
new4(s(N),M) :- ~~new5(N,M).~~ new2(N,M).
new4(0,2).
~~new5(s(N),M) :- new3(N,M).~~
```

Further Improvements (2)

Eliminating transient clauses by unfolding, we get:

```
det_win(0,M).
det_win(s(s(N)),M) :- new3(N,M).

new3(0,1).
new3(s(N),M) :- new4(N,M).
new4(0,2).
new4(s(s(N)),M) :- new3(N,M).
```

Conclusions

Automatic derivation of a winning strategy

http://www.iasi.cnr.it/~proietti/system.html

Invariants are captured by folding steps

(see R. Backhouse et al.)

Computation of the next move in constant (or log) time after an initial linear cost (see R. Bird: Loopless Functional Algorithms)

For the future: - more experiments

- other classes of games