Transparent Synchronous Dataflow

A programming paradigm for dataflow graphs

Steven W.T. Cheung, Dan R. Ghica, Koko Muroya

CHARACTERISTICS

transparent (implicit lifting)
higher-order
imperative
parallel
deadlock free
memory-safe
space-safe
time-safe
support feedbacks safely
allow to be modified in run-time

QUICK EXAMPLE

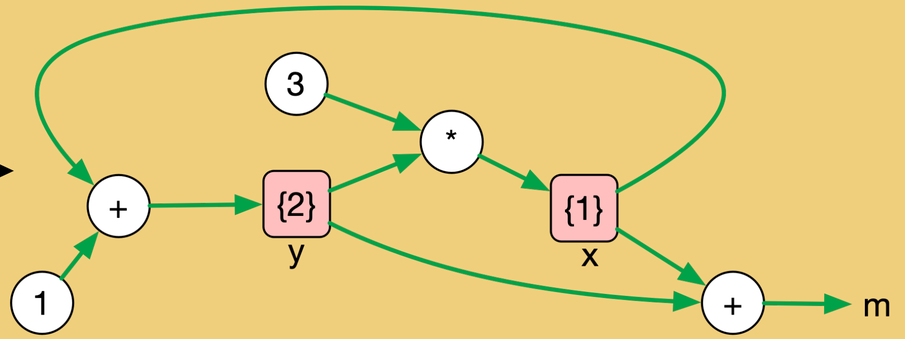
```
let x = ref 1 in
let y = ref 2 in
let m = deref y + deref x in
y <~ 1 + deref x;
x <~ deref y * 3;
let b = step in
m
```

create a cell

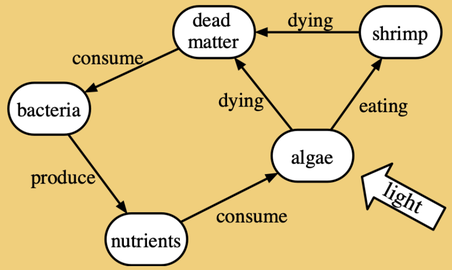
create dependencies

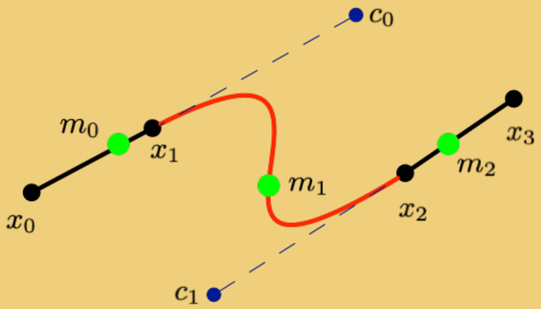
rewrite the graph

synchronous step



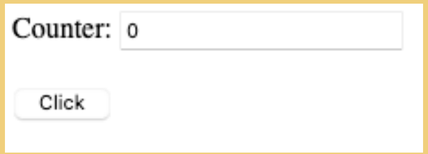
APPLICATIONS





systems modelling and optimisation

```
let acc = ref 0 in
acc <~ acc + buttonClick;
$textbox.text = acc
```



GUI programming

LINKS

Visualiser:
https://cwtsteven.github.io/TSD-visual/

OCaml implementation:
https://github.com/cwtsteven/TSD

UNIVERSITY OF BIRMINGHAM