Playing with Time and Space in Circuits and Programs

Gérard Berry
Collège de France
Chaire Algorithmes, machines et langages

PPS, 24/04/2013
Agenda

1. 2-adic numbers and space / time exchange in synchronous circuits

2. Never determinize non-deterministic automata !

3. Use hierarchical automata for another exponential gain in space and timing optimization
Agenda

1. 2-adic numbers and space / time exchange in synchronous circuits

2. Never determinize non-deterministic automata!

3. Use hierarchical automata for another exponential gain in space and timing optimization
Source of the 2-adic Part

2-adic Numbers (Hensel, ~1900)

• R is a completion de Q. Is it the only one?
  No: p-adic numbers for p prime
  infinite numbers written low-order bits first

• Beautiful, but physical? cf. Matière à Pensée, p. 32
  Alain Connes / JP Changeux

• Jean Vuillemin: 2-adiques integers are the right model
  of arithmetic digital circuits
  Let us create their physics!

2-adic numbers unify computable arithmetic
with Boolean logic
$2\mathbb{Z}$: the Ring of 2-adic Numbers

$x = 2x_0x_1x_2 \ldots$ low-order bits first

operations $+$ and $\times$ from left to right

\[
\begin{align*}
0 &= 2000000... = 2(0) \\
1 &= 2100000... = 21(0) \\
2 &= 2010000... = 201(0)
\end{align*}
\]

\[
\begin{align*}
-1 &= 2111111... = 2(1) \\
-2 &= 2011111... = 20(1)
\end{align*}
\]

\[
\begin{align*}
x &= 2101010... = 2(10) \\
&= 2100000... + 2001010... \\
&= 1 + 4x
\end{align*}
\]

\[
\begin{align*}
x &= -1/3 \\
y &= 2x \\
or \quad x + y &= -1
\end{align*}
\]

\[
\begin{align*}
y &= -2/3
\end{align*}
\]
\( \mathbb{Z}_2 : \text{the Ring of } 2\text{-adic Numbers} \)

\[ \pm \frac{p}{q} \text{ exists for all integer } p, q \text{ iff } q \text{ est odd (cf. Euclide)} \]

1/2 does not exist because \( x_0 + x_0 \) cannot have value 1

No order compatible with the operations

\[ \not\exists -1 \leq 0 \leq 1 \]
\( \mathbb{Z}_2 \) as a Boolean Algebra

- 2-adic \( x \) seen as the set \( \{ i \mid x_i = 1 \} \)
  
  Example: \(-1/3 = 2^{101010...} = \{ i \mid i \text{ even} \} \)

- Pointwise Boolean operations:
  \[
  x \land y \quad x \lor y \quad \neg x \\
  (x \land y)_n = x_n \land y_n \quad \text{etc.}
  \]

- Fundamental arithmetico-logical equality:
  \[
  x + \neg x = -1
  \]
  \[
  2_1^{100011...} \quad 2_2^{011100...} \\
  \underline{+} \quad \underline{+} \\
  2_1^{111111...}
  \]
Cantor Metric Space

\[ d(x,x) = 0 \]
\[ d(x,y) = 2^{-n} \quad n \text{ minimal s.t. } x_n \neq y_n \]

Example: \( d(201111..., 201101...) = 1/8 \)

**Lemma**: \( 2Z \) is ultrametric:

\[ d(x,z) \leq \max(d(x,y), d(y,z)) \]

\[ d(x,z) = \min(d(x,y), d(y,z)) \]
Cantor Metric Space

- Open set basis: finite prefixes
  \[ x_0 x_1 \ldots x_n \rightarrow \{ 2^0 x_0 x_1 \ldots x_n y_0 y_1 \ldots y_n \ldots \mid y \in 2\mathbb{Z} \} \]

ex.: open set for \(2^{10010}\)

- Compact – very different from reals!
Lemma: \( f: \mathbb{Z}_2 \rightarrow \mathbb{Z}_2 \) continuous iff \( f(x)_n \) depends on a finite number of \( x_m \)

Continuity = preservation of information finiteness
Synchronous Functions

$x \rightarrow f(x)$

$x_0x_1\ldots x_n \rightarrow 0x_0x_1\ldots x_n$
**Synchronous and Contracting Functions**

- **Theorem**: \( f : 2\mathbb{Z} \to 2\mathbb{Z} \) is synchronous iff \( f(x) \) only depends on \( x_0 x_1 \ldots x_n \), i.e., iff \( f \) is contracting

\[ \forall x, y. \; d(f(x), f(y)) \leq d(x, y) \]

- **Definition**: \( f : 2\mathbb{Z} \to 2\mathbb{Z} \) synchronous iff computable by a synchronous circuit (with finite or infinite memory)

**Preuve**: « only if » trivial,
for « if » see SDD construction later on
Moore Circuits and Strict Contraction

• A **Moore** synchronous circuit is such that any wire between an input and an output traverses a register.

- **Theorem**: A function \( f : 2\mathbb{Z} \to 2\mathbb{Z} \) is **strictly contracting** iff \( f(x)_n \) only depends on \( x_0x_1\ldots x_{n-1} \)

\[ \forall \ x, y. \ d(f(x), f(y)) < d(x, y) \]

• A function \( f : 2\mathbb{Z} \to 2\mathbb{Z} \) is strictly contracting iff \( f(x)_n \) only depends on \( x_0x_1\ldots x_{n-1} \)

\[ \forall \ x, y. \ d(f(x), f(y)) < d(x, y) \]
Feedbacks in Moore Circuits are Legal

\[ x \rightarrow \text{Moore Circuit} \rightarrow f(x) \]
**Feedbacks in Moore Circuits are Legal**

\[
\forall x,y. \quad d(f(x), f(y)) < d(x, y)
\]

\[
\iff
\]

\[
\forall x,y. \quad d(f(x), f(y)) < 0.6 \cdot d(x, y)
\]

---

**Banach theorem:** any Lipschitzian function over a compact set has a unique fixpoint
Full Adder

\[ s = a \text{ xor } b \text{ xor } c \]
\[ r = (a \text{ and } b) \text{ or } (b \text{ and } c) \text{ or } (c \text{ and } a) \]
Addition in Space

\[ r_0 = 0 \]

\[ a_0 \]
\[ b_0 \]
\[ a_1 \]
\[ b_1 \]
\[ a_2 \]
\[ b_2 \]
\[ \infty \]

\[ s_0 = a_0 + b_0 \]
\[ s_1 = a_1 + b_1 \]
\[ s_2 = a_2 + b_2 \]

but within infinite time!

continuity:
cut at \( n \) bits
for \( n \) output bits

\[ x \cdot 2^n = x \mod 2^n \]

\[ s \cdot 2^{n+1} = a \cdot 2^n + b \cdot 2^n \]
Full Adder

\[ s = a \ xor \ b \ xor \ c \]
\[ r = (a \ and \ b) \ or \ (b \ and \ c) \ or \ (c \ and \ a) \]

\[ a + b + c = s + 2r \]
Basic 2-adic Operators

\[ a + b + c = s + 2r \]

\[ \begin{array}{c}
2x_0x_1...x_n... \\
\Downarrow \\
2^0x_0x_1...x_n... \\
\hline \\
2^1x_0x_1...x_n... \\
\Downarrow \\
1 + 2x \\
\end{array} \]

\[ \begin{array}{c}
x \\
\Downarrow \\
2x \\
\hline \\
1 + 2x \\
\end{array} \]
Addition and Subtraction Over Time

\[ a + b + 2r = s + 2r \]
\[ s = a + b \]

same equation as over space!

\[ a + \overline{b} + 1 + 2r = s + 2r \]
\[ b + \overline{b} = -1 \]
\[ \overline{b} + 1 = -b \]
\[ a - b = s \]
\[ s = a - b \]
Mixed Space / Time Addition

\[ \begin{align*}
    x \circ y &= 2x_0y_0x_1y_1 \\
    a &= a_e \circ a_o \\
    b &= b_e \circ b_o \\
    s &= s_e \circ s_o \\
    s &= a + b
\end{align*} \]

still the same equation!

Same source code for any space / time tradeoff
Stereo Addition

Alternates 2 additions over time (even / odd bits)

\[ s_e \circ s_o = (a_e + b_e) \circ (a_o + b_o) \]

Stereo = left / right channels
Addition and Subtraction Over Time

\[
\begin{align*}
    & a \quad \text{+} \quad s \\
    & b
\end{align*}
\]

\[
\begin{align*}
    & a \quad \text{+} \quad s \\
    & b
\end{align*}
\]

\[
\begin{align*}
    & a \quad \text{+} \quad s \\
    & b
\end{align*}
\]

\[
\begin{align*}
    & a \quad \text{+} \quad s \\
    & b
\end{align*}
\]
Multiplication and division by a constant

proof: \( x + 2x = 3x \)

proof: \( y = x - 2y \)

division only by odd integers!
Quasi-inverse

\[ y = \frac{1}{1 - 2x} \]

\[ y - 2xy = 1 \]

\[ y = 1 + 2xy \]

contracting \(\Rightarrow\) synchronous but infinite memory

(cf. SDD construction)
Quasi Square Root

\[ y = \sqrt{1+8x} \]

\[ y = 1 + 4z \]
\[ y^2 = 1 + 8z + 16z^2 \]
\[ z = x - 2z^2 \]
\[ y^2 = 1 + 8x - 16z^2 + 16z^2 \]

... but tells us nothing about bit transformations!
Spatio-Temporal Decomposition of $f$ Synchronous

$f \cdot 0 = \text{first bit output by } f \text{ for inputs } 0...$

$f \cdot 1 = ... 1...$

$f \cdot w = \text{last bit output by } f \text{ for the finite word } w$

$f^0 = 0\text{-predictor} : f^0 \cdot w = f \cdot (w0) \text{ for any word } w$

$f^1 = 1\text{-predictor} : f^1 \cdot w = f \cdot (w1)$

$f^u = u\text{-predictor} : f^u \cdot w = f \cdot (wu) \text{ for any words } w, u$
Automaton of $x \rightarrow 3x$
**Predictor** $0$ of $x \rightarrow 3x$
SDD Decomposition Step

\[ f(x) = \text{mux}(x, f \cdot 1 + 2f^1(x), f \cdot 0 + 2f^0(x)) \]
SDD Space/Time Normal Form of $f$

Truth-table in space and time
ultra-fast : critical path = one mux
Half of the bits disappear at each cycle
Shared SDD of $f$ with Finite Memory

$f$ finite memory $\Rightarrow$ finitely many distinct predictors $f^u$

$f$ with $n$ registers $\Rightarrow$ SDD($f$) may have $2^n$ registers
From continuous functions to circuits

f continuous but not synchronous:
• over space: trivial if infinite space
• over time: expand the time

2-adic number: \(<\text{value}, \text{validity}>\)

\[\begin{align*}
0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & \ldots \\
0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & \ldots \\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & \ldots \\
\end{align*}\]

Theorem: every continuous function can be realized by a synchronous circuit with validity
Trace of a Synchronous Function

\[ \text{Tr}(f) = \_2 f \cdot 0 f \cdot 1 f \cdot 00 f \cdot 01 f \cdot 10 f \cdot 11 f \cdot 000 f \cdot 001 \ldots \]
\[ = f \cdot 0 + 2 f \cdot 1 + 4 (\text{Tr}(f^0) \odot \text{Tr}(f^1)) \]

Application of a trace \( \text{Tr}(f) \) to an argument \( x \)
is continuous \( \Rightarrow \) \( \lambda \)-calculus?

Power series over \( \mathbb{Z}/2\mathbb{Z} \) : 
\[ S(f) = \sum_n \text{Tr}(f)_n z^n \]

**Theorem:** \( f : \mathbb{Z}_2 \rightarrow \mathbb{Z}_2 \) synchronous has finite memory
iff \( S(f) \) is algebraic over \( \mathbb{Z}/2\mathbb{Z} \)
Theorem (Van der Porten) : if $f$ has finite memory, then the real number

$$0, f\cdot0 f\cdot1 f\cdot00 f\cdot01 f\cdot10 f\cdot11 f\cdot000 f\cdot001 \ldots$$

is either rational or transcendental

Almost any finite automaton generates a transcendental number!

*Automatic Sequences: Theory, Applications, Generalizations*

Jean-Paul Allouche et Jeffrey Shallit

Cambridge University Press (21 juillet 2003)
Conclusion

Thanks to Jean Vuillemin

• 2-adic numbers are the good model of arithmetic synchronous circuits (only?)

• the 2-adic metric, continuity, and synchronism are fundamental notions to explore further

• The structure of the predictor space is largely unknown

• The relation between continuous functions and validity-circuits remains to be studied (λ-calculus?)
Agenda

1. 2-adic numbers and space / time exchange in synchronous circuits

2. Never determinize non-deterministic automata!

3. Use hierarchical automata for another exponential gain in space and timing optimization
From Deterministic Automata to Circuits

\[(ab+b)^*ba\]

1-hot encoding (only one \(r_i\) to 1)

size explosion!
The Non-Deterministic Case

\((ab+b)^*ba\)

no size explosion \(\Rightarrow\) much better!
Electrical Subset Construction

\[(ab+b)^*ba\]
Electrical Subset Construction

\[(ab+b)^*ba\]

\[\text{(tick!)}\]
Electrical Subset Construction

\[(ab+b)^*ba\]
Electrical Subset Construction

$$(ab+b)^*ba$$

Graph with states $s_0$, $s_1$, $s_2$, and $s_3$, with transitions labeled by $b$ and $a$.

Tick!
(ab+b)*ba

Electrical Subset Construction
Electrical Subset Construction
Electrical Subset Construction

\[(ab+b)^*ba\]
Electrical Subset Construction

\[(ab+b)^*ba\]

Diagram:

- States: \(s_0, s_1, s_2, s_3\)
- Transitions:
  - \(s_0 \rightarrow s_1\) on \(b\)
  - \(s_1 \rightarrow s_0\) on \(a\)
  - \(s_0 \rightarrow s_2\) on \(b\)
  - \(s_2 \rightarrow s_3\) on \(a\)
  - \(s_3 \rightarrow s_0\) on \(a\)

- Output:
  - \(\text{tick!}\)
  - \(\text{abba}\)
  - \(\text{ok}\)
Esterel v7 implementation

module Autom :
input a, b ;
output ok ;
local {r0, r1, r2, r3} : reg ;
refine r0 : init true ;
sustain {
    next r0 <= (r0 or r1) and b ,
    next r1 <= r0 and a ,
    next r2 <= r0 and b ,
    next r3 <= r2 and a ,
    ok <= next r3 }
end module

Compiled into C, C++, VHDL, Verilog, etc.
1. 2-adic numbers and space / time exchange in synchronous circuits

2. Never determinize non-deterministic automata!

3. Use hierarchical automata for another exponential gain in space and timing optimization
The ABRO Example

Emit O as soon as A and B have arrived
Reset behavior each time R is received

Memory write
R : Request
A : Address
B : Data
O : Write
Esterel : Linear Program

loop
  abort
  \{ await A || await B \};
  emit O;
  halt
when R
end loop

copies = residuals !
Esterel = sharing residuals
Hierarchical synchronous concurrent automata
(Synchronous Statecharts)

```
loop
  abort
    { await A || await B };
  emit O;
  halt
when R
end loop
```
The ABCRO example

Flat automaton
Hierarchical automaton

source: l'oreille cassée, Hergé
From Esterel to Circuits

Circuit = constructive proof network
loop
  { await A || await B } ;
emit O
each R

parallel threads => independent groups
sequence => group-hot
1-hot: 4 bits
log: 2 bits
group-hot: 3 bits – better scaling