Short-Circuit Logic

Alban Ponse

section Theory of Computer Science

Informatics Institute

University of Amsterdam

www.science.uva.nl/~alban/

November 4, 2013

1. Introduction

Imperative programming: let P and Q be program fragments and consider

$$\text{if } (a \mathbin{\&\&} (b \mid\mid c)) \text{ then } (P) \text{ else } (Q)$$

QUESTION: Wrt conditions as above, which logical laws are valid?

For example, is left-distributivity of $\&\&$ over $\mid\mid$, that is

$$x \mathbin{\&\&} (y \mid\mid z) = (x \mathbin{\&\&} y) \mid\mid (x \mathbin{\&\&} z)$$

a valid law for conditions in imperative programming?

Assume `(i==k)` is an instruction that tests whether program variable `i` has value $k \in \mathbb{Z}$

(a) *Suppose* the mentioned left-distributivity is valid

(b) *Suppose* the assignment `[i:=i+1]` when evaluated as a test yields `true` if `i` has (initial) value 2, then

`[i:=i+1] && ((i==2) || (i==3))` yields `true` and

`([i:=i+1] && (i==2)) || ([i:=i+1] && (i==3))` yields `false`

$\Rightarrow$ (a) and (b) are contradictory

$\Rightarrow$ (a) is not true here because (b) is ($\pm$) common programming practice

Different forms of sequential evaluation of `&&` (and `||`) exist:

Suppose `i` has (initial) value 2, then

`((i==2) || [i:=i+1]) && (i==2)`

evaluates to

`true`	with *short-circuit* evaluation (SCE)
`false`	with *full* evaluation (all atoms are evaluated)

We first restrict to SCE:

The semantics of Boolean operators in programming languages in which the second argument is only executed/evaluated if the first argument does not suffice to determine the value of the expression

QUESTION: which logic characterizes SCE?

2. Short-Circuit Logic, Case 1: atoms only

A truth table inspired semantics with ingredients:

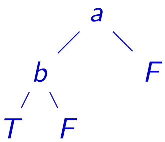
1. A, a countable set of atoms (atomic propositions) $a, b, \ldots$
2. $SProp$, the set of sequential propositional statements (closed terms) over the signature

$$\{ \circ\!\!\wedge \,, \overset{\circ}{\vee} \,, \neg, a \mid a \in A\}$$

where $\circ\!\!\wedge$ and $\overset{\circ}{\vee}$ are directed versions of conjunction and disjunction, respectively, that prescribe SCE (cf. `&&` and `||`, respectively)

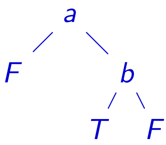
Notation: T for `true` and F for `false`

All possible evaluations of $a \,{}_{\circ}\!\!\wedge\, b$ are characterized by the following evaluation tree:

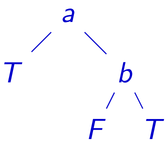


1. Branches descending to the left of an internal node indicate that the node is evaluated T and to the right that it yielded F
2. An evaluation is a complete path
3. The leaf in which an evaluation ends represents the (final) value of that evaluation

Two more examples of evaluation trees that illustrate *negation* and *left-sequential disjunction* $\mathbin{{}^{\circ}\!\!\vee}$:



Evalution tree of $\neg a \mathbin{{}_{\circ}\!\!\wedge} b$



Evalution tree of $a \mathbin{{}^{\circ}\!\!\vee} \neg b$

Given some evaluation tree X, an evaluation can be represented by

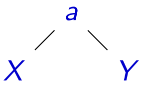
$$(\sigma, B)$$

with $\sigma \in (A \cup \{T, F\})^*$ and $B \in \{T, F\}$, where $(\sigma\restriction_A)B$ is a full path in X
Example: $(aFbF, T)$ is the rightmost evaluation of $a \mathbin{{}^{\circ}\!\!\vee} \neg b$ in the rightmost tree above

$\mathbb{T}$: evaluation trees over A with leaves in $\{T, F\}$ is defined inductively:

$$T \in \mathbb{T}, \quad F \in \mathbb{T}, \quad (X \trianglelefteq a \trianglerighteq Y) \in \mathbb{T} \quad \text{for any } X, Y \in \mathbb{T} \text{ and } a \in A$$

$X \trianglelefteq a \trianglerighteq Y$ can be represented by



Leaf replacement of T with Y and F with Z in X is denoted

$$X[T \mapsto Y, F \mapsto Z]$$

and is defined inductively by

$$T[T \mapsto Y, F \mapsto Z] = Y$$

$$F[T \mapsto Y, F \mapsto Z] = Z$$

$$(X \trianglelefteq a \trianglerighteq X')[T \mapsto Y, F \mapsto Z] = X[T \mapsto Y, F \mapsto Z] \trianglelefteq a \trianglerighteq X'[T \mapsto Y, F \mapsto Z]$$

Convention: no listing of identities inside the brackets, e.g.,

$$X[F \mapsto Z] = X[T \mapsto T, F \mapsto Z]$$

$\Rightarrow$ Terminology and notation to formally define the interpretation of SCE-terms as evaluation trees in $\mathbb{T}$ (i.e., the set of all full binary trees with nodes in A and leaves in $\{T, F\}$)

$\Rightarrow$ Define the unary Short-Circuit Evaluation function

$$se : SProp \to \mathbb{T}$$

as follows, where $a \in A$:

$$\begin{aligned}
se(a) &= T \trianglelefteq a \trianglerighteq F \\
se(\neg P) &= se(P)[T \mapsto F, F \mapsto T] \\
se(P \mathbin{\circ\!\!\wedge} Q) &= se(P)[T \mapsto se(Q)] \\
se(P \mathbin{\circ\!\!\vee} Q) &= se(P)[F \mapsto se(Q)]
\end{aligned}$$

Thm 0. *se*-equality for *SProp* has this equational axiomatization:

$$\neg\neg x = x$$

$$x \mathbin{\circ\!\!\vee} y = \neg(\neg x \mathbin{\circ\!\!\wedge} \neg y)$$

$$(x \mathbin{\circ\!\!\wedge} y) \mathbin{\circ\!\!\wedge} z = x \mathbin{\circ\!\!\wedge} (y \mathbin{\circ\!\!\wedge} z)$$

That is, for all $P, Q \in SProp$,

$$E \vdash P = Q \iff se(P) = se(Q)$$

Proof. Soundness ($\Longrightarrow$) is trivial; completeness ($\Longleftarrow$) is less...

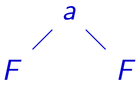
(Note: axiomatization defines left-sequential duality)

3. Short-Circuit Logic, Case 2: adding T and F as constants to $SProp$

$$\begin{aligned}
se(T) &= T\\
se(F) &= F\\
se(a) &= T \trianglelefteq a \trianglerighteq F\\
se(\neg P) &= se(P)[T \mapsto F, F \mapsto T]\\
se(P \mathbin{{}_{\circ}\!\wedge} Q) &= se(P)[T \mapsto se(Q)]\\
se(P \mathbin{{}^{\circ}\!\vee} Q) &= se(P)[F \mapsto se(Q)]
\end{aligned}$$

Example:

$$se(a \mathbin{{}_{\circ}\!\wedge} F) = F \trianglelefteq a \trianglerighteq F =$$



NOTE: the three axioms mentioned are sound under this extension and *se*-equality remains a congruence

Four obvious axioms (and their duals):

$$\neg T = F \qquad\qquad \neg F = T$$

$$T \leftand x = x \qquad\qquad F \leftor x = x$$

$$x \leftand T = x \qquad\qquad x \leftor F = x$$

$$F \leftand x = F \qquad\qquad T \leftor x = T$$

There are many more non-trivial identifications, e.g., for all propositions P,

$$se(P \leftand F) = se(\neg P \leftand F)$$

Three more axioms:

$$x \mathbin{{}_{\circ}\!\wedge} F = \neg x \mathbin{{}_{\circ}\!\wedge} F$$

$$(x \mathbin{{}_{\circ}\!\wedge} F) \mathbin{{}^{\circ}\!\vee} y = (x \mathbin{{}^{\circ}\!\vee} T) \mathbin{{}_{\circ}\!\wedge} y$$

(here, y will always be evaluated)

$$(x \mathbin{{}_{\circ}\!\wedge} y) \mathbin{{}^{\circ}\!\vee} (z \mathbin{{}_{\circ}\!\wedge} F) = (x \mathbin{{}^{\circ}\!\vee} (z \mathbin{{}_{\circ}\!\wedge} F)) \mathbin{{}_{\circ}\!\wedge} (y \mathbin{{}^{\circ}\!\vee} (z \mathbin{{}_{\circ}\!\wedge} F))$$

(here, $\mathbin{{}^{\circ}\!\vee}$ right-distributes over $\mathbin{{}_{\circ}\!\wedge}$ whenever its right-argument yields F)

Thm 1. (Daan Staudt, 2012) The set E containing the ten listed axioms is an equational axiomatization of SCE for $SProp$: for all $P, Q \in SProp$,

$$E \vdash P = Q \iff se(P) = se(Q)$$

Proof.

$\Longrightarrow$: (Soundness) trivial

$\Longleftarrow$: (Normal forms + decomposition properties of se-trees) $\Rightarrow$ inverse of normalization function

(this part of the proof takes 20^+ pages)

4. Conditional Propositions (and proposition algebra)

Hoare's ternary conditional operator (1985) $y \triangleleft x \triangleright z$ resembles

if (x) then (y) else (z)

where if (..) then (..) else (..) is used as a propositional connective

Hoare's equational laws that characterize Propositional Logic include the equational basis of *free valuation congruence*, which we named *CP* (for Conditional Propositions):

$$x \triangleleft T \triangleright y = x$$
$$x \triangleleft F \triangleright y = y$$
$$T \triangleleft x \triangleright F = x$$
$$x \triangleleft (y \triangleleft z \triangleright u) \triangleright v = (x \triangleleft y \triangleright v) \triangleleft z \triangleright (x \triangleleft u \triangleright v)$$

SCE is the only reasonable kind of evaluation for conditional propositions:

Let *CPprop* be the set of conditional propositional statements over the signature

$$\{\_ \triangleleft \_ \triangleright \_, T, F, a \mid a \in A\}$$

Extend the function $se : CPprop \to \mathbb{T}$ by

$$se(P \triangleleft Q \triangleright R) = se(Q)[T \mapsto se(P), F \mapsto se(R)]$$

Thm 2. CP is an equational axiomatization of SCE as adapted here, that is, for all $P, Q \in CPprop$,

$$CP \vdash P = Q \iff se(P) = se(Q)$$

Proof. $\Longrightarrow$ is trivial

$\Longleftarrow$ immediately follows from the proof in our paper on Proposition Algebra [Bergstra and Ponse (2011)] (that employs valuation varieties)

All of $\circ\!\!\wedge$, $\circ\!\!\vee$, $\neg$ are definable in CP:

$$\neg x = F \triangleleft x \triangleright T$$

$$x \mathbin{\circ\!\!\wedge} y = y \triangleleft x \triangleright F$$

$$x \mathbin{\circ\!\!\vee} y = T \triangleleft x \triangleright y$$

... but $\_ \triangleleft \_ \triangleright \_$ is not expressible with $\circ\!\!\wedge$, $\circ\!\!\vee$, $\neg$ only (for example, $se(a \triangleleft a \triangleright a)$ contains four traces with atom length 2 etc.)

In CP extended with these connectives, one easily derives $x \triangleleft \neg y \triangleright z = x \triangleleft (F \triangleleft y \triangleright T) \triangleright z = (x \triangleleft F \triangleright z) \triangleleft y \triangleright (x \triangleleft T \triangleright z) = z \triangleleft y \triangleright x$, and thus

$$\begin{aligned}
\neg(\neg x \mathbin{\circ\!\!\wedge} \neg y) &= F \triangleleft (\neg y \triangleleft \neg x \triangleright F) \triangleright T \\
&= (F \triangleleft \neg y \triangleright T) \triangleleft \neg x \triangleright (F \triangleleft F \triangleright T) \\
&= T \triangleleft x \triangleright y \\
&= x \mathbin{\circ\!\!\vee} y
\end{aligned}$$

5. Several Short-Circuit Logics

A generic definition: a Short-circuit logic is

- a logic that implies all consequences of CP that can be expressed with $\circ\!\!\wedge$, $\circ\!\!\vee$, $\neg$ and $a \in A$

- or, more precisely, a logic that implies all consequences of the *module expression SCL* defined by

$$\begin{aligned} SCL = \{T, \neg, \circ\!\!\wedge\} \,\square\, (CP & \\ & + \langle \neg x = F \triangleleft x \triangleright T \rangle \\ & + \langle x \circ\!\!\wedge y = y \triangleleft x \triangleright F \rangle) \end{aligned}$$

Now F can in SCL be used as a shorthand for $\neg T$ because

$$CP + \langle \neg x = F \triangleleft x \triangleright T \rangle \vdash \neg T = F \triangleleft T \triangleright F = F$$

(and $\circ\!\!\vee$ is also definable)

$\Rightarrow$ All axioms in E can easily be derived from CP and the definitions of $\circ\!\!\wedge$, $\circ\!\!\vee$, $\neg$ in CP i.e., from the module SCL

Example:
$$\begin{aligned} \neg x \circ\!\!\wedge F &= F \triangleleft (F \triangleleft x \triangleright T) \triangleright F \\ &= (F \triangleleft F \triangleright F) \triangleleft x \triangleright (F \triangleleft T \triangleright F) \\ &= F \triangleleft x \triangleright F \\ &= x \circ\!\!\wedge F \end{aligned}$$

FSCL (Free short-circuit logic) is the short-circuit logic that implies no other consequences than those of CP

NOTE: *FSCL* is the *least identifying* short-circuit logic we define

(As a consequence,)

Thm 1. (Daan Staudt, 2012) The set E containing the ten listed axioms is an equational axiomatization of *FSCL*

A more identifying SCL:

Write $CP_{rp}(A)$ (Repetition-proof CP) for CP extended with these axiom schemes ($a \in A$):

$$(x \triangleleft a \triangleright y) \triangleleft a \triangleright z = (x \triangleleft a \triangleright x) \triangleleft a \triangleright z$$

$$x \triangleleft a \triangleright (y \triangleleft a \triangleright z) = x \triangleleft a \triangleright (z \triangleleft a \triangleright z)$$

$RPSCL$ (Repetition-proof short-circuit logic) is the short-circuit logic that implies no other consequences than those of $CP_{rp}(A)$

i.e., no other consequences than those of the module expression

$$\begin{aligned}\{T, \neg, \circ\!\!\wedge, a \mid a \in A\} \,\square\, (CP_{rp}(A) & \\ &+ \langle \neg x = F \triangleleft x \triangleright T \rangle \\ &+ \langle x \circ\!\!\wedge y = y \triangleleft x \triangleright F \rangle)\end{aligned}$$

Axioms for $RPSCL$ include those in E and for $a \in A$,

$$a \mathbin{{}_{\circ}\!\!\wedge} (a \mathbin{{}^{\circ}\!\!\vee} x) = a \mathbin{{}_{\circ}\!\!\wedge} a$$

$$a \mathbin{{}^{\circ}\!\!\vee} (a \mathbin{{}_{\circ}\!\!\wedge} x) = a \mathbin{{}^{\circ}\!\!\vee} a$$

Properties of T and F as defined in E can be mimicked in context, and imply more axioms, e.g.,

$$(a \mathbin{{}^{\circ}\!\!\vee} \neg a) \mathbin{{}_{\circ}\!\!\wedge} x = (\neg a \mathbin{{}_{\circ}\!\!\wedge} a) \mathbin{{}^{\circ}\!\!\vee} x \qquad (T \mathbin{{}_{\circ}\!\!\wedge} x = F \mathbin{{}^{\circ}\!\!\vee} x)$$

$$(\neg a \mathbin{{}^{\circ}\!\!\vee} a) \mathbin{{}_{\circ}\!\!\wedge} x = (a \mathbin{{}_{\circ}\!\!\wedge} \neg a) \mathbin{{}^{\circ}\!\!\vee} x$$

$$(a \mathbin{{}_{\circ}\!\!\wedge} \neg a) \mathbin{{}_{\circ}\!\!\wedge} x = a \mathbin{{}_{\circ}\!\!\wedge} \neg a \qquad (F \mathbin{{}_{\circ}\!\!\wedge} x = F)$$

QUESTION: Has E a finite/countable extension that is an equational axiomatization of $RPSCL$?

Example on *RPSCL*:

- arithmetic expressions over Naturals (or Int's)

- each atom is either test or assignment

- assignments as conditions (Boolean evaluation) yield T

Then, $RPSCL \vdash a \mathbin{{}_{\circ}\!\wedge} (a \mathbin{{}^{\circ}\!\vee} x) = a \mathbin{{}_{\circ}\!\wedge} (a \mathbin{{}^{\circ}\!\vee} y)$, e.g.,

$$\texttt{[i:=i+1]} \mathbin{{}_{\circ}\!\wedge} (\texttt{[i:=i+1]} \mathbin{{}^{\circ}\!\vee} \texttt{(i==2)})$$

$$\texttt{[i:=i+1]} \mathbin{{}_{\circ}\!\wedge} (\texttt{[i:=i+1]} \mathbin{{}^{\circ}\!\vee} \texttt{(i==0)})$$

both evaluate to T and have the same (side) effect

[Wortel (2011)]: Case study on an "extension" of *Dynamic Logic*
(extension?: in DL, each program can be turned into to a test)

RPSCL does not model the equivalence discussed in the Introduction (imperative programming), even not if the atoms in conditions are restricted to assignments and pure tests (like `(i==2)`)

not: In practice ("Expression languages"), the Boolean evaluation of an assignment is that of the assigned value (Int's: F for 0, and T otherwise):

While $RPSCL \vdash a \wedge^{\circ} (a \,{}^{\circ}\!\vee\, x) = a \wedge^{\circ} (a \,{}^{\circ}\!\vee\, y)$, we find

$$\texttt{[i:=i+1]} \mathbin{{}_{\circ}\!\wedge} (\texttt{[i:=i+1]} \mathbin{{}^{\circ}\!\vee} \texttt{(i==2)}) \text{ yields } \begin{cases} F & \text{if } \texttt{i} \text{ equals } -2, \\ T & \text{otherwise,} \end{cases}$$

but $\texttt{[i:=i+1]} \mathbin{{}_{\circ}\!\wedge} (\texttt{[i:=i+1]} \mathbin{{}^{\circ}\!\vee} \texttt{(i==0)})$ always yields T

Write CP_{st} (Static CP) for CP extended with these axioms:

$$T \triangleleft x \triangleright y = T \triangleleft y \triangleright x$$

$$(x \triangleleft y \triangleright z) \triangleleft y \triangleright F = x \triangleleft y \triangleright F$$

that is, "$x \mathbin{\circ\!\!\!\vee} y = y \mathbin{\circ\!\!\!\vee} x$" and "positive contraction", respectively

(equivalent extensions of CP that define CP_{st} are recorded)

$SSCL$ (Static short-circuit logic) is the short-circuit logic that implies no other consequences than those of CP_{st}

i.e., no other consequences than those of the module expression

$$\begin{aligned}\{T, \neg, \mathbin{\circ\!\!\!\wedge}\} \,\square\, (CP_{st} \\ &+ \langle \neg x = F \triangleleft x \triangleright T \rangle \\ &+ \langle x \mathbin{\circ\!\!\!\wedge} y = y \triangleleft x \triangleright F \rangle)\end{aligned}$$

Thm 3. *SSCL* (and sequential propositional logic) is axiomatized by

$$T = x \mathbin{^{\circ}\!\vee} \neg x$$
$$F = \neg T$$

$$x \mathbin{_{\circ}\!\wedge} y = y \mathbin{_{\circ}\!\wedge} x$$
$$x \mathbin{_{\circ}\!\wedge} (y \mathbin{^{\circ}\!\vee} z) = (x \mathbin{_{\circ}\!\wedge} y) \mathbin{^{\circ}\!\vee} (x \mathbin{_{\circ}\!\wedge} z)$$
$$x \mathbin{_{\circ}\!\wedge} (y \mathbin{^{\circ}\!\vee} \neg y) = x$$

+ the duals of the last two axioms (cf. [Sioson (1964)])

Now T and F are definable, and only now: in all valuation semantics that identify less, this is not so

Sequential propositional logic applies to the case of conditions composed from atoms that have no side effects (pure tests)

6. Conclusions and Future Work

1. Some more *SCL*s were defined, and for one of those we have an equational axiomatization (Memorizing *SCL*)

2. Based on the proposition algebras we introduced, more *SCL*s can be defined; many *SCL*s are natural and simple and deserve attention

3. A next step: consider a partition of the set A of atoms into *side effect free* atoms (like `(i==3)`) and the rest (like `(i:=3)`, finer partitions are possible); wrt *RPSCL* an initial study was done by Wortel (2011)

   (NOTE: in this case, an atom like `(3==3)` can play the role of T)

4. *Full left-sequential evaluation* is also relevant (x & y in programming), and was studied by Blok (2011) and Staudt (2012):

$$x \mathbin{\overset{\bullet}{\wedge}} y = (x \mathbin{\overset{\circ}{\vee}} (y \mathbin{\overset{\circ}{\wedge}} F)) \mathbin{\overset{\circ}{\wedge}} y$$

Less expressive; complete axiomatizations were found;
both families of connectives and item 3 provide setting for general analysis (normalization or simplification of conditions)

5. "cand" is sometimes used for $\overset{\circ}{\wedge}$ in a setting with SCE, and && is often used in programming

6. SCE is also named *minimal evaluation*, *McCarthy evaluation* or *shortcut evaluation*

7. Last version short-circuit logic paper (Bergstra, Ponse, Staudt):
   http://arxiv.org/abs/1010.3674 (v4, 12 March 2013)
   (More info at my home page > Research)

8. The notation $\mathbin{{}_\circ\!\wedge}$, $\mathbin{{}^\circ\!\vee}$ was introduced in
   J.A. Bergstra, I. Bethke, and P.H. Rodenburg (1995).
   A propositional logic with 4 values: true, false, divergent and meaningless.
   Journal of Applied Non-Classical Logics, 5(2):199-218.

9. More references:

J.A. Bergstra and A. Ponse (2011). Proposition Algebra.
ACM Transactions on Computational Logic, 12(3), Article 21 (36 pp).

C.A.R. Hoare (1985). A couple of novelties in the propositional calculus.
Zeitschr. f . Math. Logik und Grundlagen d. Math., 31(2), 173-178.

F.M. Sioson (1964). Equational Bases of Boolean Algebras.
Journal of Symbolic Logic 29 (3):115-124.