SEMANTIC INDEPENDENCE*

Eike BEST**

Gesellschaft für Mathematik und Datenverarbeitung, Institut für Methodische Grundlagen, Schloß Birlinghoven, D-5205 St. Augustin 1, FRG

Christian LENGAUER***

Department of Computer Sciences, The University of Texas at Austin, Taylor Hall 2.124, Austin, TX 78712-1188, USA

Communicated by A. Pnueli
Received September 1987
Revised February 1989

Abstract. Semantic independence is defined as a generalization of the Bernstein condition which implies that two assignments are concurrently executable. It is shown that semantically independent programs are commutative and concurrently implementable.

1. Introduction

The research described in this paper originates from two questions. Firstly, how can the property that two PASCAL-like programs are independent (in the sense that the results of one of them are unimportant to the other) be defined formally? Secondly, how can data dependencies be completely characterized? These questions are important for the exploration of potential concurrency in an algorithm.

As a simple case, we may consider two assignments $\alpha_1$ and $\alpha_2$, such as in the following example:

\begin{verbatim}
var x, y: {0, 1};
\alpha_1: x := x \oplus 1
\alpha_2: y := y \ominus 1
\end{verbatim}

* Part of this work has been presented at the Annual Workshop on Application and Theory of Petri Nets, Zaragoza (Spain) [3].
** Present address: Institut für Informatik, Universität Hildesheim, D-3200 Hildesheim, FRG.
*** Funded in part by the National Science Foundation under Contract DCR-8610427.

In our examples, we always specify the value sets of the variables by means of a declaration of the form \texttt{var} \(x, y, \ldots\) : \texttt{value set}. The value set of a variable \(x\) will be denoted by \(Val(x)\). If the value set is of the form \(\{0, \ldots, m\}\) for \(m \geq 1\), then the operators \(\ominus\) and \(\ominus\) denote cyclic addition and subtraction, respectively. If the value set is \(\{0, 1\}\), then the operators \(\&\), \(\lor\), \(\neg\), etc. have their usual Boolean meaning of "and," "or," "negation," etc.; we identify 0 with \texttt{false} and 1 with \texttt{true}, as usual.
In this example, \( \alpha_1 \) and \( \alpha_2 \) are variable-disjoint. Hence, intuitively, they "do not interfere with each other." Clearly, there are no data dependencies between \( \alpha_1 \) and \( \alpha_2 \). Also, \( \alpha_1 \) and \( \alpha_2 \) could be executed fully concurrently, and as another consequence, \( \alpha_1 \) and \( \alpha_2 \) are commutative, i.e., executing them in any order yields the same final result.

Similar remarks are true if \( \alpha_1 \) and \( \alpha_2 \) are such that any common variables appear only at the right-hand sides of the assignments, i.e., are read only. For instance, consider example (2):

\[
\begin{align*}
\text{var } x, y, z & : \{0, 1\}; \\
\alpha_1 & : x := z \\
\alpha_2 & : y := z
\end{align*}
\]

Again, even though \( \alpha_1 \) and \( \alpha_2 \) depend on the common variable \( z \), \( z \) is read only and therefore, \( \alpha_1 \) and \( \alpha_2 \) "do not interfere with each other": neither does \( \alpha_1 \) depend on \( \alpha_2 \) nor vice versa. Again, \( \alpha_1 \) and \( \alpha_2 \) can be executed concurrently, and they commute.

Both in example (1) and in example (2), the two assignments \( \alpha_1 \) and \( \alpha_2 \) satisfy the so-called Bernstein condition \([1, 2, 10]\) which requires that any common variables do not appear on the left-hand side of the assignments. (We will define this condition more precisely below.) However, there exist more subtle examples of assignments which one would like to call independent even though they do not satisfy the Bernstein condition. One such example has been given in \([4]\):

\[
\begin{align*}
\text{var } x, y & : \{0, 1, 2, 3\}; \\
\alpha_1 & : x := 2 \times (y \mod 2) + (x \mod 2) \\
\alpha_2 & : y := 2 \times (x \mod 2) + (y \mod 2)
\end{align*}
\]

This example can be analyzed as follows. The value set of both \( x \) and \( y \) is \( \{0, 1, 2, 3\} \), whence \( x \) and \( y \) can both be represented as two-bit variables. The expression \((x \mod 2)\) denotes bit 0 of \( x \) and the expression \((y \mod 2)\) denotes bit 0 of \( y \). Then \( \alpha_1 \) has the effect of transferring bit 0 of \( y \) into bit 1 of \( x \) (leaving bit 0 of \( x \) unchanged), while \( \alpha_2 \) has the effect of transferring bit 0 of \( x \) into bit 1 of \( y \) (leaving bit 0 of \( y \) unchanged), as shown schematically in Fig. 1. Since the bits affected by \( \alpha_1 \) are
disjoint from the bits affected by $\alpha_2$, it is meaningful to call $\alpha_1$ and $\alpha_2$ independent, even though they fail to satisfy the Bernstein condition. We choose the terminology "semantic independence" to stress the fact that we wish to capture a property which may not be readily apparent from the syntactic structure of $\alpha_1$ and $\alpha_2$. The reader may easily check that commutativity holds for example (3) as well.

Example (3) may also be analyzed in a slightly different way. We may define new binary variables $x_0$, $x_1$, $y_0$, and $y_1$ by means of the following set of equations:

$$\begin{align*}
\text{var } & x_0, x_1, y_0, y_1: \{0, 1\}; \\
x_0 & = x \mod 2; \\
x_1 & = x \div 2; \\
y_0 & = y \mod 2; \\
y_1 & = y \div 2. 
\end{align*}$$

(3a)

These equations define a mapping of pairs $(x, y)$ to quadruples $(x_0, x_1, y_0, y_1)$. The inverse mapping is defined by:

$$\begin{align*}
x & = 2 \ast x_1 + x_0; \\
y & = 2 \ast y_1 + y_0. 
\end{align*}$$

(3b)

That is, $(x, y)$ and $(x_0, x_1, y_0, y_1)$ are related one-to-one. In terms of the new variables, example (3) can be equivalently rewritten in the following form:

$$\begin{align*}
\alpha'_1: & \quad x_1 := y_0 \\
\alpha'_2: & \quad y_1 := x_0
\end{align*}$$

(3')

From this, the independence of $\alpha'_1$ and $\alpha'_2$ (and hence of $\alpha_1$ and $\alpha_2$) is evident.

In example (3), the two assignments fail to satisfy the Bernstein condition, but they are still special in the sense that the two left-hand sides are disjoint. One may wonder whether this is indicative of semantic independence. However, provided that one allows (as, for the sake of generality, we will) so-called multiple assignments (see [5, Section 9.2]) one may find examples with nondisjoint left-hand sides. Consider example (4) below.

$$\begin{align*}
\text{var } & x, y, z: \{0, 1\}; \\
\alpha_1: & \quad (x, y) := (x \land z, y \land (z \oplus 1)) \\
\alpha_2: & \quad (x, y) := (x \land (z \oplus 1), y \land z)
\end{align*}$$

(4)

The multiple assignment $(x, y) := (e_1, e_2)$, where $e_1, e_2$ are expressions, has the following semantics: first, $e_1$ and $e_2$ are both evaluated and then, the value of $e_1$ is assigned to $x$ and the value of $e_2$ is assigned to $y$; $x$ and $y$ must be distinct variables.
Example (4) can be analyzed as follows. One may distinguish exactly two cases: 
\( z = 0 \) and \( z = 1 \). If \( z = 0 \), then the two assignments \( \alpha_1 \) and \( \alpha_2 \) can be simplified to:

\[
\begin{align*}
\alpha_1: & \quad (x, y) := (0, y) \\
\alpha_2: & \quad (x, y) := (x, 0)
\end{align*}
\]

i.e.,

\[
\begin{align*}
\alpha_1: & \quad x := 0 \\
\alpha_2: & \quad y := 0
\end{align*}
\]

If, on the other hand, \( z = 1 \), then the two assignments can be simplified to:

\[
\begin{align*}
\alpha_1: & \quad (x, y) := (x, 0) \\
\alpha_2: & \quad (x, y) := (0, y)
\end{align*}
\]

i.e.,

\[
\begin{align*}
\alpha_1: & \quad y := 0 \\
\alpha_2: & \quad x := 0
\end{align*}
\]

In both cases, it is clear that \( \alpha_1 \) and \( \alpha_2 \) are independent, since they operate on disjoint sets of variables. Hence \( \alpha_1 \) and \( \alpha_2 \) are independent overall.

Like example (3), example (4) can also be analyzed in terms of transforming the variables. This analysis will be deferred to Section 3. Again, the reader may easily check that \( \alpha_1 \) and \( \alpha_2 \) commute in (4).

From the discussion so far, it may appear that the commutativity of \( \alpha_1 \) and \( \alpha_2 \) implies their semantic independence. This is not true, as is demonstrated by the next example:

\[
\text{var } x: \{0, 1, 2, 3\};
\]

\[
\begin{align*}
\alpha_1: & \quad x := x \oplus 1 \\
\alpha_2: & \quad x := x \oplus 2
\end{align*}
\]  \hspace{1cm} (5)

In this example, \( \alpha_1 \) and \( \alpha_2 \) commute but clearly, they are not independent of each other.

The goal of this paper is to find a definition of "semantic independence" which satisfies at least the following requirements:

(a) Examples (1)-(4) should satisfy semantic independence but example (5) should not.
(b) The Bernstein condition should imply semantic independence.
(c) Semantic independence should imply commutativity.
(d) The semantic independence of two assignments should allow for their implementation and execution on disjoint pieces of memory (in a sense to be made precise).
(e) The definition of semantic independence should not appeal to the particular syntactic form of an assignment or to the way in which variables are accessed.
Requirement (e) seems to be essential in order to capture situations as in examples (3) and (4); moreover, it also has the welcome consequence that the definition of semantic independence is general and can be applied to any pair of programs, not just to assignments.

2. Semantic independence

We will use input/output relations on the state space as our principal means to describe the semantics of program fragments (like assignments).

Definition 2.1. Let

\[ \text{var } x_1 : \text{Val}(x_1), \ldots, x_m : \text{Val}(x_m); \quad (m \geq 1) \]

be a declaration. The state space \( S \) associated with it is defined as

\[ S = \text{Val}(x_1) \times \cdots \times \text{Val}(x_m). \]

Thus, for instance, in (1) we have

\[ S = \text{Val}(x) \times \text{Val}(y) = \{0, 1\} \times \{0, 1\} = \{00, 01, 10, 11\} \]

(we write 00, 01, etc. short for \((x = 0, y = 0), (x = 0, y = 1), \text{etc} \). In (5), we simply have

\[ S = \text{Val}(x) = \{0, 1, 2, 3\}. \]

We will denote the value of a variable \( x \) in the state \( s \in S \) by

\[ \text{val}(x, s). \]

Further, we will denote the set of values of an expression \( e \) in the state \( s \in S \) by

\[ \text{Val}(e, s). \]

Definition 2.2. Let

\[ \text{var } x_1 : \text{Val}(x_1), \ldots, x_m : \text{Val}(x_m); \quad (m \geq 1) \]

be a declaration, \( S \) its associated state space and

\[ \alpha : (y_1, \ldots, y_k) := (e_1, \ldots, e_k) \]

a (multiple) assignment such that \( y_1, \ldots, y_k \) are mutually distinct variables and \( \alpha \) contains only variables from \( x_1, \ldots, x_m \). Then we associate with \( \alpha \) a meaning relation

---

3 We allow here nondeterministic expressions that may have more than one value in a given state, as well as undefined expressions that may have no value in a given state. We need to be so general for one of the theorems which we wish to prove later. However, in practice (as well as in all of our examples), \( e \) will satisfy the property of having a well-defined value in each state.
with the following definition:

\[(s', s) \in m(\alpha)\]

iff

(i) \(\forall j, 1 \leq j \leq k: val(y_j, s) \in Val(\epsilon_j, s');\) and

(ii) \(\forall x \in (\{x_1, \ldots, x_m\}\setminus\{y_1, \ldots, y_k\}): val(x, s) = val(x, s').\)

Definition 2.2 reflects the usual meaning of the assignment: 2.2(i) means that in the final state \(s\), each \(y_j\) contains any one of the values that could be taken by \(\epsilon_j\) in the initial state \(s'\); 2.2(ii) means that the values of all other variables remain unchanged. The reader may easily check that if all \(\epsilon_j\) are deterministic and always yield a well-defined value in \(Val(y_j)\), then \(m(\alpha)\) is a function from \(S\) to \(S\).

To approach our definition of semantic independence, we will now reconsider examples (1) and (5) of the introduction. Figures 2 and 3 depict their state spaces and meaning relations; we use the pictorial notation defined in Notation 2.3.

Notation 2.3 (Pictorial conventions). We will use the following uniform notation:

- solid arrows denote \(m(\alpha_1)\); broken arrows denote \(m(\alpha_2)\);
- \(00\) denotes \(x = 0, y = 0\); \(01\) denotes \(x = 0, y = 1\), etc.

Our task is to determine, only by looking at the state space \(S\) and the two meaning relations \(m(\alpha_1) \subseteq S \times S\) and \(m(\alpha_2) \subseteq S \times S\) (that is, only by looking at the graphs shown in Figs. 2 and 3), whether or not \(\alpha_1\) and \(\alpha_2\) are semantically independent. The key point is to realize that every variable \(x\) induces a canonical equivalence
relation, which we will call $\rho_x$, on the state space $S$. This equivalence relation is defined by

$$(s', s) \in \rho_x \iff \text{val}(x, s') = \text{val}(x, s).$$

Clearly, $\rho_x$ is an equivalence relation and therefore, $\rho_x$ induces a partitioning of $S$. In Fig. 2, for instance, $\rho_x$ induces the partitioning $\{(00, 01), \{10, 11\}\}$ of $S$, while $\rho_y$ induces the partitioning $\{(00, 10), \{01, 11\}\}$ of $S$. Figure 4 shows the two partitionings, together with $m(\alpha_1)$ and $m(\alpha_2)$, for Fig. 2, that is, for example (1) (here we also use Notation 2.4).

**Notation 2.4 (Further pictorial conventions).** Equal shapes enclose classes of the same equivalence relation. (In the case of Fig. 4, ovals denote classes of $\rho_y$ and rectangles denote classes of $\rho_x$.)

In example (1), $\alpha_1$ and $\alpha_2$ "behave well" with respect to the two partitionings $\rho_x$ and $\rho_y$ induced by $x$ and $y$: $m(\alpha_1)$ connects only states which are in $\rho_y$, while $m(\alpha_2)$ connects states which are in $\rho_x$ (that is, $m(\alpha_1) \subseteq \rho_y$ and $m(\alpha_2) \subseteq \rho_x$). By contrast, consider example (5) in Fig. 3. There is only one variable (namely $x$), and $x$ induces the discrete partitioning, i.e., every singleton set $\{s\}$, with $s \in S$, is an equivalence class of $\rho_x$. Neither $m(\alpha_1)$ nor $m(\alpha_2)$ "behave well" with respect to $\rho_x$. Even worse, there is no way at all to partition the state space $S$ by means of two partitionings in such a way that the latter could correspond to two variables and

---

4 As is well known, equivalence relations on $S$ and partitionings of $S$ can be used interchangeably; when there is no danger of confusion, we will sometimes switch back and forth between equivalence relations and the induced partitionings, i.e., the set of their associated equivalence classes.
such that $m(\alpha_1)$ and $m(\alpha_2)$ "behave nicely" with respect to them; it is our task to capture this distinction between examples (1) and (5) precisely.

We shall invert the connection between variables and partitionings. According to requirement (e) listed in the introduction, the definition of semantic independence should not depend on a particular predetermined way in which $S$ is built up from variables. Hence we will speak of partitionings of $S$ rather than of variables, keeping the possible interpretation of partitionings as variables in mind as a motivation. Our projected definition of semantic independence can be phrased as follows: $\alpha_1$ and $\alpha_2$ are semantically independent if there exist two partitionings $\rho_1, \rho_2$ of $S$ (with some special properties) such that $m(\alpha_1)$ and $m(\alpha_2)$ "behave nicely" with respect to $\rho_1$ and $\rho_2$ (this must still be made precise), just as in the example shown in Fig. 4.

We are now ready to formalize these ideas. We shall start with an arbitrary set $S$ (the state space) and two arbitrary relations $m_1 \subseteq S \times S$ and $m_2 \subseteq S \times S$, and we shall define what it means for $m_1$ and $m_2$ to be semantically independent on $S$. To this end, we require the existence of two partitionings $\rho_1$ and $\rho_2$ which are like the partitionings induced by variables; what the latter means is captured by the next definition.

**Definition 2.5.** Let $S$ be a set and let $\rho_1 \subseteq 2^S$, $\rho_2 \subseteq 2^S$ be two partitionings of $S$.\textsuperscript{5} Then $\rho_1$ and $\rho_2$ will be called orthogonal iff

(i) $\forall X \in \rho_1 : \forall Y \in \rho_2 : |X \cap Y| \geq 1$; and
(ii) $\forall X \in \rho_1 : \forall Y \in \rho_2 : |X \cap Y| \leq 1$.

\textsuperscript{5} $2^S$ denotes the power set of $S$. 
Of course, the conjunction of 2.5(i) and 2.5(ii) simplifies to \( \forall X \in \rho_1 : \forall Y \in \rho_2 : |X \cap Y| = 1 \). But we prefer to treat conditions 2.5(i) and 2.5(ii) separately. In a later section, we shall investigate possible relaxations of Definition 2.5 by elimination of either of the two conditions. Therefore, we find it useful in proofs to be aware of the particular inequality we exploit.

One can easily deduce from Definition 2.5 that if \( S \) is a finite set and \( \rho_1, \rho_2 \) are two orthogonal partitionings of \( S \) then \( |S| = |\rho_1| \cdot |\rho_2| \) and, moreover, \( \forall X \in \rho_1 : |X| = |\rho_2| \) and \( \forall Y \in \rho_2 : |Y| = |\rho_1| \).

Orthogonality generalizes the Cartesian product: if \( S = \text{Val}(x) \times \text{Val}(y) \) for two variables \( x \) and \( y \), then \( \rho_x \) and \( \rho_y \) (the partitionings induced by \( x \) and \( y \)) are orthogonal. That is, two orthogonal partitionings have the same properties as the partitionings induced by two distinct variables. We will investigate the meaning of Definition 2.5 more closely in Section 4 below.

**Definition 2.6.** Let \( S \) be a set and let \( m_1 \subseteq S \times S \) and \( m_2 \subseteq S \times S \) be two relations on \( S \). Then \( m_1 \) and \( m_2 \) will be called *semantically independent* iff there exist two equivalence relations (or two partitionings) \( \rho_1, \rho_2 \) on \( S \) such that:

1. \( \rho_1 \) and \( \rho_2 \) are orthogonal.
2. (a) \( m_1 \subseteq \rho_2 \);
   (b) \( m_2 \subseteq \rho_1 \).
3. (a) \( \rho_2 \cap (\rho_1 \circ m_1 \circ \rho_1) \subseteq m_1 \);
   (b) \( \rho_1 \cap (\rho_2 \circ m_2 \circ \rho_2) \subseteq m_2 \).

Requirement 2.6(i) arises from the desire to regard \( \rho_1 \) and \( \rho_2 \) as distinct variables. Requirement 2.6(ii) has already been motivated: \( m_1 \) may not change the “value” of \( \rho_2 \), and \( m_2 \) may not change the “value” of \( \rho_1 \). Finally, requirement 2.6(iii) expresses that \( m_1 \) may not read (i.e., act upon) the “value” of \( \rho_2 \), and \( m_2 \) may not read (i.e., act upon) the “value” of \( \rho_1 \).

Obviously the relation of semantic independence is not transitive. However it is not hard to generalize the definition to express the mutual semantic independence of a set \( \alpha = \{m_1, \ldots, m_n\} \) of \( n \geq 2 \) relations: \( \alpha \) is a set of mutually semantically independent relations iff there are \( n \) partitionings \( \rho_i \) \((1 \leq i \leq n)\) such that

\[
\forall (X_1, \ldots, X_n) \in \rho_1 \times \cdots \times \rho_n : \left| \bigcap_{i=1}^{n} X_i \right| = 1, \quad (6)
\]

\[
\forall i, 1 \leq i \leq n : \forall j, 1 \leq j \leq n : i \neq j \Rightarrow \rho_j \cap (\rho_i \circ m_i \circ \rho_i) = m_i. \quad (7)
\]

\( ^6 \) In this definition, \( \subseteq \) denotes the inclusion of relations, \( \cap \) denotes the intersection of relations and \( \circ \) denotes relational composition.
(6) generalizes 2.6(i) and (7) generalizes 2.6(ii)-(iii). We have included this generalization for the sake of completeness, but we will not need it in the remainder of this paper.

Before proving that this definition satisfies requirements (a)-(e) listed in the introduction, we discuss a further example that demonstrates the necessity of both parts of condition 2.6(iii); see Fig. 5.

In Fig. 5, $\alpha_1$ and $\alpha_2$ should not be semantically independent since they are not even commutative. On the other hand, consider the two partitionings

$$\rho_1 = \rho_x = \{\{00, 01\}, \{10, 11\}\}$$

and

$$\rho_2 = \rho_y = \{\{00, 10\}, \{01, 11\}\}.$$  

Then $\rho_1$ and $\rho_2$ satisfy conditions (i), (ii) and (iii.a) of Definition 2.6; only condition 2.6(iii.b) is violated. A symmetrical example demonstrates the necessity of condition 2.6(iii.a).

Let us test Definition 2.6 on the more complicated example (4) of the introduction. Figure 6 repeats the example and depicts its state relations. In this example, none of the equivalence relations $\rho_x, \rho_y, \rho_z$ induced by the three variables $x, y$ and $z$ reveal the independence of $\alpha_1$ and $\alpha_2$, because within and between the equivalence

\begin{align*}
\text{var } x, y & : \{0, 1\}; \\
\alpha_1 : & x := 1 \\
\alpha_2 : & y := x \lor y
\end{align*}

Fig. 5. Illustration of the definition of semantic independence.
classes of $\rho_\alpha$, $\rho_\beta$, and $\rho_\gamma$ there are mixtures of both $m(\alpha_1)$-arrows and $m(\alpha_2)$-arrows (cf. Fig. 6). However, consider the following two equivalence relations on $S$:

$$\rho_1 = \{\{000, 010, 001, 101\}, \{100, 110, 011, 111\}\},$$
$$\rho_2 = \{\{000, 100\}, \{010, 110\}, \{001, 011\}, \{101, 111\}\}.$$

Figure 7 depicts these two relations.

As opposed to $\rho_x$, $\rho_y$, and $\rho_z$, the relations $\rho_1$ and $\rho_2$ satisfy all conditions of Definition 2.6 and hence demonstrate the semantic independence of $\alpha_1$ and $\alpha_2$. Firstly, $\rho_1$ and $\rho_2$ are orthogonal (which means that they could arise from a decomposition of $S$ into variables such that $\rho_1$ corresponds to a variable with two values and $\rho_2$ corresponds to a variable with four values). Secondly, condition 2.6(ii) is satisfied since all $m(\alpha_1)$-arrows lie within $\rho_2$-classes and all $m(\alpha_2)$-arrows lie within $\rho_1$-classes. Thirdly, condition 2.6(iii) is also satisfied, as the reader may easily check.

Let us now address the question whether Definition 2.6 satisfies requirements (a)-(e) listed in the introduction. Requirement (e) demands that the definition of semantic independence should be general, paying no respect to the particular form of the assignments $\alpha_1$, $\alpha_2$ or to the way in which $S$ is composed from variables.
This desideratum is clearly satisfied, since Definition 2.6 only assumes an arbitrary set $S$ and two arbitrary relations on $S$ to start with. Thus, the case of two assignments is really a special case; Definition 2.6 applies to any two program fragments on a common state space.

Requirement (d) needs a relatively extensive discussion and will therefore be deferred to Section 3.

Requirement (c) is satisfied due to the following theorem:

**Theorem 2.7.** Let $S$ be a set and let $m_1, m_2 \subseteq S \times S$ be two relations which are semantically independent. Then $m_1$ and $m_2$ commute, i.e.,

$$m_1 \circ m_2 = m_2 \circ m_1.$$ 

**Proof.** We show only $m_1 \circ m_2 \subseteq m_2 \circ m_1$; the proof of $m_1 \circ m_2 \supseteq m_2 \circ m_1$ follows by symmetry.

Suppose $(s', s) \in m_1 \circ m_2$; then $\exists t \in S: (s', t) \in m_1$ and $(t, s) \in m_2$. By the semantic independence of $m_1, m_2$, there are equivalence relations $\rho_1$ and $\rho_2$ on $S$ with properties of Definition 2.6. By property 2.6(ii), we have $(s', t) \in \rho_2$ and $(t, s) \in \rho_1$.

Let $X$ be the unique $\rho_1$-equivalence class containing $t$ and $s$ and $Y$ the unique $\rho_2$-equivalence class containing $s'$ and $t$ (see Fig. 8). Further, let $X'$ be the unique $\rho_1$-equivalence class containing $s'$ and $Y'$ the unique $\rho_2$-equivalence class containing...
Fig. 8. Illustration of the proof of commutativity.

s. The orthogonality of $\rho_1$, $\rho_2$ guarantees $|X' \cap Y'| \geq 1$ (see Definition 2.5(i)), and hence we may choose a state $t' \in X' \cap Y'$ (see Fig. 8). We will show that $(s', t') \in m_2$. To this end, we note that $(s', t') \in \rho_1$ and that $(s', t) \in \rho_2$, $(t, s) \in m_2$ and $(s', t') \in \rho_2^{-1} = \rho_2$, whence $(s', t') \in (\rho_2 \circ m_2 \circ \rho_2)$. Hence 2.6(iii.b) implies the desired result $(s', t') \in m_2$. Similarly, 2.6(iii.a) yields $(t', s) \in m_1$. Hence we have $(s', s) \in m_2 \circ m_1$, which was to be proved. □

Remark 2.8. (i) Example (5) shows that Theorem 2.7 cannot be inverted.

(ii) Definition 2.5(ii) has not been used in the proof of Theorem 2.7.

Requirement (b) of the introduction will be addressed by the next definition and the subsequent theorem.

Definition 2.9. Let

$$\text{var } x_1 : \text{Val}(x_1), \ldots, x_m : \text{Val}(x_m);$$

$$\alpha_1 : (y_1, \ldots, y_k) := (\delta_1, \ldots, \delta_k)$$

$$\alpha_2 : (z_1, \ldots, z_l) := (\varepsilon_1, \ldots, \varepsilon_l)$$

be two (multiple) assignments with predeclared variables.

(i) $\alpha_1$ and $\alpha_2$ satisfy the strict Bernstein condition iff none of the variables occurring in $\alpha_1$ also occurs in $\alpha_2$ (i.e., $\alpha_1$ and $\alpha_2$ are variable-disjoint, like in (1)).

(ii) $\alpha_1$ and $\alpha_2$ satisfy the weak Bernstein condition iff the variables $y_1, \ldots, y_k$ do not occur in $\alpha_2$ and the variables $z_1, \ldots, z_l$ do not occur in $\alpha_1$ (i.e., any common variables are restricted to the right-hand sides of $\alpha_1$, $\alpha_2$, as in (2)).

Clearly, if $\alpha_1$, $\alpha_2$ satisfy the strict Bernstein condition then they also satisfy the weak Bernstein condition.
Theorem 2.10. With the same notation as in Definition 2.9, suppose that $\alpha_1$ and $\alpha_2$ satisfy the weak Bernstein condition. Then $m(\alpha_1)$ and $m(\alpha_2)$ are semantically independent on

$$S = \text{Val}(x_1) \times \cdots \times \text{Val}(x_m).$$

Proof. We have to construct equivalence relations $\rho_1$, $\rho_2$ on $S$ which, together with $m(\alpha_1)$ and $m(\alpha_2)$, satisfy Definition 2.6. To achieve this, we generalize the equivalence relation $\rho_x$ induced by a variable $x$.

Let $W = \{w_1, \ldots, w_q\} \subseteq \{x_1, \ldots, x_m\}$ be any (sub)set of variables. Then $\rho_w \subseteq S \times S$ is defined as follows:

$$(s', s) \in \rho_w \iff \forall w \in W: \text{val}(w, s') = \text{val}(w, s).$$

Clearly, $\rho_w$ is an equivalence relation and $\rho_{\{x\}} = \rho_x$ for all variables $x$. Now let $V = \{y_1, \ldots, y_k\}$ (that is, the variables on the left-hand side of $\alpha_i$) and let $W = \{x_1, \ldots, x_m\} \setminus V$ (that is, the remaining variables which, by the Bernstein condition, include $\{z_1, \ldots, z_l\}$). Define $\rho_1 = \rho_v$ and $\rho_2 = \rho_w$; we claim that $\rho_1$ and $\rho_2$ satisfy Definition 2.6.

To show 2.6(i), let $X$ be any equivalence class of $\rho_1$; by definition of $\rho_1$, the value of each variable in $V$ is constant across all states in $X$. Let $Y$ be any equivalence class of $\rho_2$; similarly, $Y$ determines a unique value for each variable in $W$. Hence $X \cap Y$ determines a unique value for each variable in $\{x_1, \ldots, x_m\}$, which is the same as saying that $X \cap Y$ determines a unique state in $S$. Therefore $|X \cap Y| = 1$.

To show 2.6(ii.a), i.e., $m(\alpha_1) \subseteq \rho_2$, suppose that $(s', s) \in m(\alpha_1)$. By Definition 2.2(ii), the values of the variables in $W$ are equal in $s'$ and in $s$, which implies $(s', s) \in \rho_2$. Definition 2.6(ii.b) can be shown analogously. Finally, we show 2.6(iii.a), i.e., $\rho_2 \cap (\rho_1 \circ m(\alpha_1) \circ \rho_1) \subseteq m(\alpha_1)$. Pick any $(s', s) \in \rho_2 \cap (\rho_1 \circ m(\alpha_1) \circ \rho_1)$. To prove that $(s', s) \in m(\alpha_1)$ we must demonstrate the properties of Definition 2.2 with respect to $m(\alpha_1)$. $(s', s) \in \rho_2$ means that $s'$ and $s$ do not differ in their $W$-variables, which implies 2.2(ii).

$(s', s) \in \rho_1 \circ m(\alpha_1) \circ \rho_1$ means that $\exists t', t: (s', t') \in \rho_1, (t', t) \in m(\alpha_1)$ and $(t, s) \in \rho_1$, which implies 2.2(i) since $t'$ and $t$ are related by $m(\alpha_1)$. Requirement 2.6(iii.b) can be shown analogously.

Remark 2.11. Examples (3) and (4) show that Theorem 2.10 cannot be inverted.

Finally, we check item (a) of the list of requirements given in the introduction. To this end, we need to apply Definition 2.6 to the examples (1)-(5) of the introduction. Examples (1) and (2) exhibit semantic independence by Theorem 2.10 since the weak Bernstein condition is satisfied. Example (4) also satisfies the requirements of semantic independence, as has been demonstrated earlier. In example (5), it is not possible to find $\rho_1$, $\rho_2$ which satisfy Definition 2.6: any attempt of establishing 2.6(ii) contradicts the orthogonality of $\rho_1$, $\rho_2$ (even regardless of 2.6(iii)).
This leaves example (3) to be discussed. It is quite easy to see that (3) also satisfies Definition 2.6—define two equivalence relations $\rho_1$ and $\rho_2$ as follows:

$$(s', s) \in \rho_1 \iff val(x \mod 2, s') = val(x \mod 2, s)$$

$$\land val(y \div 2, s') = val(y \div 2, s),$$

$$(x', s) \in \rho_2 \iff val(x \div 2, s') = val(x \div 2, s)$$

$$\land val(y \mod 2, s') = val(y \mod 2, s).$$

What is more, this analysis of example (3) does not depend on the size of the value sets of $x$ and $y$. Say, we allow the four values \{0, 1, 2, 3\} for $x$ and the set of natural numbers for $y$. Then $\alpha_1$ and $\alpha_2$ as in (3) are still semantically independent, with the same $\rho_1$ and $\rho_2$. Hence Definition 2.6 is more general than the bitwise decomposition that Fig. 1 illustrates.

### 3. Disjoint implementation

The semantic independence of two assignments should allow for their implementation and execution on disjoint pieces of memory, as demanded by requirement (d) listed in the introduction. This property is obvious only in (1) where $\alpha_1$ and $\alpha_2$ operate on disjoint sets of variables. Already for example (2), it is not obvious whether $\alpha_1$ and $\alpha_2$ can be implemented disjointly, and it is even less clear for (3) and (4).

In this section, we will show that all semantically independent assignments can be implemented disjointly, and we will also exhibit variable transformations as one way of achieving such an implementation. The guiding idea is that two assignments can be implemented disjointly provided that they are equivalent to two other assignments which satisfy the strict Bernstein condition 2.9(i), in the same way as example (3) in the introduction is equivalent to (3'). The next theorem formalizes this idea.

**Theorem 3.1.** Let $S$ be an arbitrary set (the state space) and let $m_1 \subseteq S \times S$ and $m_2 \subseteq S \times S$ be two relations on $S$. Then the following are equivalent:

- $m_1$ and $m_2$ are semantically independent.
- There exist two variables $V$ and $W$, with value sets $Val(V)$ and $Val(W)$, respectively, such that with the state space

$$S' = Val(V) \times Val(W)$$

the following is true: there exists a bijection

$$\beta : S \to S'$$

and two relations

$$\mu \subseteq Val(V) \times Val(V) \quad \text{and} \quad \nu \subseteq Val(W) \times Val(W)$$
such that with the two assignments

\[ \alpha'_1: V := \mu(V) \quad \text{and} \quad \alpha'_2: W := \nu(W) \]

and their meaning relations

\[ m'_1 = m(\alpha'_1) \subseteq S' \times S' \quad \text{and} \quad m'_2 = m(\alpha'_2) \subseteq S' \times S' \]

the following two conditions are satisfied for all \( s', s \in S \):

(i) \((s', s) \in m_1 \iff (\beta(s'), \beta(s)) \in m'_1, \)

(ii) \((s', s) \in m_2 \iff (\beta(s'), \beta(s)) \in m'_2. \)

The form of \( \alpha'_1 \) and \( \alpha'_2 \) indicates that \( \alpha'_1 \) and \( \alpha'_2 \) satisfy the strict Bernstein condition: \( \alpha'_1 \) depends only on \( V \) and changes only \( V \), while \( \alpha'_2 \) depends only on \( W \) and changes only \( W \). The bijection \( \beta \) identifies the states of \( (Y_1, L_Y) \) with the states of \( (V_1, L_V) \), while the last condition of the theorem states that \( \alpha_1 \) is equivalent to \( \alpha'_1 \) modulo this identification, and likewise for \( \alpha_2, \alpha'_2 \). To check the theorem for examples (3) and (3') of the introduction, the reader may take \( V \) to be the composite variable \((x_1, y_0)\) and \( W \) to be the composite variable \((x_0, y_1)\); then \( \alpha'_1, \alpha'_2 \) become:

\[ \alpha'_1: (x_1, y_0) := (y_0, y_0) \quad \text{(operating only on \( V \))} \]
\[ \alpha'_2: (x_0, y_1) := (x_0, x_0) \quad \text{(operating only on \( W \))}. \]

**Proof of Theorem 3.1.**

\((\Rightarrow)\) The proof essentially works by formalizing the connection between variables and orthogonal equivalence relations mentioned informally in Section 2 to motivate Definition 2.6. We will take as \( V \) the equivalence relation \( \rho_1 \) and as \( \text{Val}(V) \) the equivalence classes of \( \rho_1 \), and similarly for \( W \) and \( \rho_2 \); that \( \rho_1 \) and \( \rho_2 \) exist is assured by the semantic independence of \( m_1 \) and \( m_2 \).

Thus, we assume \( \rho_1 \) and \( \rho_2 \) with the properties of Definition 2.6 to be given and we define:

\[ V = \rho_1, \]
\[ \text{Val}(V) = \{ X | X \text{ is equivalence class of } \rho_1 \}, \]
\[ W = \rho_2, \]
\[ \text{Val}(W) = \{ Y | Y \text{ is equivalence class of } \rho_2 \}, \]
\[ S' = \text{Val}(V) \times \text{Val}(W). \]

Given these definitions, we will now define a bijection \( \beta \) between \( S \) and \( S' \). We define \( \beta \) as a relation on \( S \times S' \) and then show that \( \beta \) is indeed a bijection. Let \( s \in S \) and \( (X, Y) \in S' \) (that is, \( X \in \text{Val}(V) \) and \( Y \in \text{Val}(W) \)). Then we define:

\[ (s, (X, Y)) \in \beta \iff s \in X \cap Y. \]

To show that \( \beta \) is a bijection we need to prove four properties:

(a) \( \beta \) is total, i.e., \( \text{dom}(\beta) = S \).
(b) $\beta$ is a function, i.e., $\beta^{-1} \circ \beta \subseteq \text{id}|_S$.
(c) $\beta$ is surjective, i.e., $\text{cod}(\beta) = S'$.
(d) $\beta$ is injective, i.e., $\beta \circ \beta^{-1} \subseteq \text{id}|_S$.

The totality (a) of $\beta$ follows from the fact that every state $s \in S$ is contained in an equivalence class of $\rho_1$ and an equivalence class of $\rho_2$. Property (b) follows from the fact that the equivalence classes containing $s$ are unique. To show the surjectivity of $\beta$, consider any pair $(X, Y) \in S'$, i.e., $X \in \text{Val}(V)$ and $Y \in \text{Val}(W)$. The orthogonality of $\rho_1$ and $\rho_2$ ensures $|X \cap Y| \neq 1$ (see Definition 2.5(i)), and hence $\exists s \in S$: $(s, (X, Y)) \in \beta$, i.e., $\text{cod}(\beta) = S'$. To show the injectivity (d) of $\beta$, suppose $(s, (X, Y)) \in \beta$ and $(s', (X, Y)) \in \beta$; then by the definition of $\beta$, $s \in X \cap Y$ and $s' \in X \cap Y$ and by Definition 2.5(ii) of orthogonality, $|X \cap Y| \neq 1$, and hence, $s = s'$.

Next we define appropriate relations $\mu \subseteq \text{Val}(V) \times \text{Val}(V)$ and $\nu \subseteq \text{Val}(W) \times \text{Val}(W)$ as follows:

\[(X', X) \in \mu \iff \exists s' \in X' \exists s \in X: (s', s) \in m_1 \]

and

\[(Y', Y) \in \nu \iff \exists t' \in Y' \exists t \in Y: (t', t) \in m_2.\]

It remains to show that conditions (i) and (ii) of the theorem are satisfied. We will first show

\[\forall s', s \in S: (s', s) \in m_1 \Leftrightarrow (\beta(s'), \beta(s)) \in m'_1.\]

To prove $(\Rightarrow)$, assume $s', s \in S$ and $(s', s) \in m_1$. Let $X'$ (and $Y'$) be the $\rho_1$-equivalence class (the $\rho_2$-equivalence class, respectively) that contains $s'$; let $X$ (and $Y$) be the $\rho_1$-equivalence class (the $\rho_2$-equivalence class, respectively) that contains $s$. Since $s' \in X' \cap Y'$ and $s \in X \cap Y$, we have $(X', Y') = \beta(s')$ and $(X, Y) = \beta(s)$. What remains to be proved is that $((X', Y'), (X, Y)) \in m'_1$. To this end, it suffices to show that the value of $W$ does not change (i.e., that $Y' = Y$) and that the value of $V$ changes in accordance with $\mu$ (i.e., that $(X', X) \in \mu$). But $(X', X) \in \mu$ is immediate from the definition of $\mu$, since $s' \in X'$, $s \in X$ and $(s', s) \in m_1$. Also, Definition 2.6(ii.a) states that $m_1 \subseteq \rho_2$; hence we know $(s', s) \in \rho_2$ and can conclude $Y' = Y$ since $s' \in X'$, $s \in Y$ and both $Y$ and $Y'$ are $\rho_2$-equivalence classes. This finishes the proof of $(\Rightarrow)$.

To prove $(\Leftarrow)$, assume $s', s \in S$, $\beta(s') = (X', Y')$, $\beta(s) = (X, Y)$ and $((X', Y'), (X, Y)) \in m'_1$; we wish to prove $(s', s) \in m_1$. First, $((X', Y'), (X, Y)) \in m'_1$ implies that $(X', X) \in \mu$ and $Y' = Y$. Then, $\beta(s') = (X', Y')$ means that $s' \in X' \cap Y'$, and $\beta(s) = (X, Y)$ means that $s \in X \cap Y$.

Further, $(X', X) \in \mu$ means that $\exists t' \in X' \exists t \in X: (t', t) \in m_1$ (see Fig. 9). We know $(s', t') \in \rho_1$, $(t', t) \in m_1$ and $(t, s) \in \rho_1$, i.e., $(s', s) \in (\rho_1 \circ m_1 \circ \rho_1)$. Also, $(s', s) \in \rho_2$ since $s' \in Y$ and $s \in Y$. Hence by 2.6(iii.a), the desired result $(s', s) \in m_1$ follows, ending the proof of $(\Leftarrow)$.

7 At this point, we need to be general and allow nondeterministic expressions as the right-hand sides of an assignment. If $m_1$ and $m_2$ are total functions, then so are $\mu$ and $\nu$. 
Condition (ii), i.e., the equivalence

$$\forall s', s \in S: (s', s) \in m_2 \iff (\beta(s'), \beta(s)) \in m'_2$$

can be proved similarly, using 2.6(ii.b) and 2.6(iii.b) instead of 2.6(ii.a) and 2.6(iii.a).

(\iff) We assume the existence of $V, W, \beta, \mu, \nu$ with the properties (i) and (ii) stated in the theorem, and we define two appropriate equivalence relations $\rho_1$ and $\rho_2$ as follows:

$$(s', s) \in \rho_1 \iff (\beta(s'), \beta(s)) \in \rho_v,$$

$$(s', s) \in \rho_2 \iff (\beta(s'), \beta(s)) \in \rho_w,$$

where $\rho_v$ and $\rho_w$ are the canonical equivalence relations induced on $S' = \text{Val}(V) \times \text{Val}(W)$ by the variables $V$ and $W$; $\rho_1$ and $\rho_2$ inherit the property of being equivalence relations from $\rho_v$ and $\rho_w$.

To prove Definition 2.6(i), we note that the orthogonality of $\rho_1$ and $\rho_2$ follows immediately from the orthogonality of $\rho_v$ and $\rho_w$ and from the fact that $\beta$ is a bijection.

To prove Definition 2.6(ii), consider first the inclusion $m_1 \subseteq \rho_2$. Pick $s', s \in S$ such that $(s', s) \in m_1$. By condition (i)(\Rightarrow), $((\beta(s'), \beta(s)) \in m(\alpha'_1)$. By the definition of $\alpha'_1$, $\text{val}(W, \beta(s')) = \text{val}(W, \beta(s))$ and hence, $(\beta(s'), \beta(s)) \in \rho_w$. By the definition of $\rho_2$, $(s', s) \in \rho_2$. The inclusion $m_2 \subseteq \rho_1$ can be proved symmetrically.

To prove Definition 2.6(iii), consider first the inclusion $\rho_2 \cap (\rho_1 \circ m_1 \circ \rho_1) \subseteq m_1$. Pick $s', s \in S$ such that

$$(s', s) \in \rho_2 \cap (\rho_1 \circ m_1 \circ \rho_1).$$

By $(s', s) \in \rho_2$ and the definition of $\rho_2$, $(\beta(s'), \beta(s)) \in \rho_w$. By $(s', s) \in \rho_1 \circ m_1 \circ \rho_1$, we may find $t', t \in S$ such that $(s', t') \in \rho_1$, $(t', t) \in m_1$ and $(t, s) \in \rho_1$. By the definition of $\rho_1$, $(\beta(s'), \beta(t')) \in \rho_v$ and $(\beta(t), \beta(s)) \in \rho_v$. By condition (i)(\Rightarrow), $(\beta(t'), \beta(t)) \in m(\alpha'_1)$. As in the proof of Theorem 2.10, we may now conclude that $(\beta(s'), \beta(s)) \in m(\alpha'_1)$. Using condition (i)(\iff), $(s', s) \in m_1$ follows.

The inclusion $\rho_1 \cap (\rho_2 \circ m_2 \circ \rho_2) \subseteq m_2$ follows by symmetry. \qed
Remark 3.2. (i) The last part of the proof of Theorem 3.1(⇒) uses the same argument as the proof of Theorem 2.7. This should not be surprising, since the commutativity of $m_1$ and $m_2$ could also be deduced from Theorem 3.1 and the (evident) commutativity of two assignments that satisfy the strict Bernstein condition.

(ii) In the proof of Theorem 3.1(⇒), Definition 2.5(ii) has been used only in order to derive the injectivity of $\beta$; in the proof of Theorem 3.1(⇐), the injectivity of $\beta$ is sufficient to prove Definition 2.5(ii).

Often, the bijection $\beta$ which exists by virtue of Theorem 3.1 can be expressed nicely in terms of a set of equations which transform the variables occurring in $\alpha_1$, $\alpha_2$ into the new variables occurring in $\alpha'_1$, $\alpha'_2$. Such a transformation has already been specified for example (3) by means of equations (3a) and (3b) in the introduction. In the remainder of this section, we will give analogous transformations for examples (2) and (4) of the introduction.

Together with Theorem 2.10, Theorem 3.1 implies that any $\alpha_1$, $\alpha_2$ which satisfy the weak Bernstein condition can be transformed into $\alpha'_1$, $\alpha'_2$ which satisfy the strict Bernstein condition. We will use example (2) (which is reproduced in Fig. 10) as a case in point to illustrate this fact.

Let $\rho_1$ and $\rho_2$ be given as shown in Fig. 10, that is, formally:

$$(s', s) \in \rho_1 \iff \text{val}(x \oplus z, s') = \text{val}(x \oplus z, s)$$

and

$$(s', s) \in \rho_2 \iff \text{val}(y, s') = \text{val}(y, s) \land \text{val}(z, s') = \text{val}(z, s)$$

(i.e., $\rho_2 = \rho_{\{y, z\}}$, but $\rho_1 \neq \rho_\alpha$). Then we have $|\rho_1| = 2$, that is, $\rho_1$ can be represented by a variable $V$ with two values, say $\text{Val}(V) = \{0, 1\}$, and similarly, $|\rho_2| = 4$, that is, $\rho_2$ can be represented by a variable $W$ with four values, say $\text{Val}(W) = \{0, 1, 2, 3\}$. Suppose that the values of $V$ and $W$ are assigned to the equivalence classes of $\rho_1$ and $\rho_2$ as shown in Fig. 10. Then the following transformation equations between $(x, y, z)$ and $(V, W)$ can be deduced:

\[
\text{var } V: \{0, 1\}, \ W: \{0, 1, 2, 3\};\\
V = (x \oplus z) \quad (\in \{0, 1\})\\
W = ((2 \ast z) \oplus y) \quad (\in \{0, 1, 2, 3\})
\] (8)

Clearly, triples $(x, y, z)$ and pairs $(V, W)$ correspond to each other bijectively (though we refrain from giving the inverse transformation explicitly). Via the transformation given by (8), the two assignments $\alpha_1$: $x := z$ and $\alpha_2$: $y := z$ are transformed into the following assignments:

\[
\text{var } V: \{0, 1\}, \ W: \{0, 1, 2, 3\};\\
\alpha'_1: \quad V := 0\\
\alpha'_2: \quad W := 3 \ast (W \div 2)
\] (2')

As can be seen, example (2') satisfies the strict Bernstein condition, and $\alpha'_1$, $\alpha'_2$ are equivalent to $\alpha_1$, $\alpha_2$ modulo the bijection defined by (8).
As our last example of this section, we deal (albeit more briefly) with example (4). We represent the two partitionings shown in Fig. 7 by two variables $V$ and $W$ with $\text{Val}(V) = \{0, 1\}$ and $\text{Val}(W) = \{0, 1, 2, 3\}$. Assigning values to equivalence classes as shown in Fig. 7 yields the transformation equations

$$V = x \land (z \oplus 1) + y \land z$$

and

$$W = 2 \ast z + x \land z + y \land (z \oplus 1)$$

and the transformed pair of assignments

$$\text{var } V: \{0, 1\}, \ W: \{0, 1, 2, 3\};$$

$$\alpha_1: \ V := 0$$

$$\alpha_2: \ W := 2 \ast (W \div 2)$$

(9)

(4')
Again, example (4') satisfies the strict Bernstein condition and is equivalent to example (4) modulo the bijection \( \beta \) defined by (9).

4. Generalizations

One of the problems in this research has been to define semantic independence neither too tightly (so that (3) and (4) could be treated) nor too loosely (so that counterexamples could not be found). We can pose two separate questions about Definition 2.6:

(i) Is it too weak?
(ii) Is it too strong?

This section addresses these two questions.

In order to answer question (i) positively, we have to find two programs which satisfy Definition 2.6 and which one would not like to call semantically independent. According to Theorem 3.1(\( \Rightarrow \)), any such pair of programs can be implemented on disjoint memory. The implementation function is guaranteed to be a bijection, i.e., the implementation is an exact model of the given programs. Therefore, this theorem makes us quite confident that such an example does not exist, i.e., that our definition of semantic independence is indeed strong enough.

In the remainder of this section we address the other question, i.e.: Do there exist programs which one would like to call semantically independent even though they do not satisfy Definition 2.6? In order to approach this question, let us have a closer look at the various parts of Definition 2.6. Property 2.6(ii) is, in our opinion, well motivated. Property 2.6(iii) can also hardly be relaxed, as the example discussed in Fig. 5 shows. Hence we concentrate our discussion on property 2.6(i), i.e., orthogonality. We will discuss what happens if this property is weakened.

Let us first consider property 2.5(ii), i.e., the second half of orthogonality:

\[
\forall X \in \rho_1 : \forall Y \in \rho_2 : |X \cap Y| \leq 1.
\]

This property has not been used in the proof of Theorem 2.7. If it is simply omitted then the resulting weakened definition of semantic independence still implies commutativity. Hence, one may ask whether Definition 2.5(ii) is a consequence of the other parts of Definition 2.6. The example displayed in Fig. 11 shows that this is not so, even if one places severe restrictions on \( m_1 \) and \( m_2 \). It demonstrates that even if the relations \( m_1 \) and \( m_2 \) are assumed to be functions, property 2.5(ii) is not a consequence of the other parts of Definition 2.6. Figure 11 is a pictorial representation of the following example:

Define \( S = (\{0, 1, 2\} \times \{0, 1, 2\}) \cup \{-1, -1\} \).

\[\begin{align*}
\text{var } z &: S; \\
\alpha_1 &: z := f_1(z) \\
\alpha_2 &: z := f_2(z)
\end{align*}\]
where $f_1 : S \rightarrow S$ and $f_2 : S \rightarrow S$ are defined as follows: for $(z_1, z_2) \in S$,

$$f_1(z_1, z_2) = \begin{cases} 
(z_1 + 2, z_2), & \text{if } z_1 = 0, \\
(z_1 + 1, z_2), & \text{if } z_1 = 1, \\
(z_1 - 2, z_2), & \text{if } z_1 = 2, \\
(2, 1), & \text{if } (z_1, z_2) = (-1, -1);
\end{cases}$$

$$f_2(z_1, z_2) = \begin{cases} 
(z_1, z_2 + 2), & \text{if } z_2 = 0, \\
(z_1, z_2 - 1), & \text{if } z_2 = 1, \\
(z_1, z_2 - 2), & \text{if } z_2 = 2, \\
(1, 0), & \text{if } (z_1, z_2) = (-1, -1).
\end{cases}$$

Moreover, define $\rho_1 = \{X_0, X_1, X_2\}$ and $\rho_2 = \{Y_0, Y_1, Y_2\}$ with:

$$X_0 = \{00, 01, 02\}, \quad X_1 = \{01, 11, 12, -1 - 1\}, \quad X_2 = \{20, 21, 22\},$$

$$Y_0 = \{00, 10, 20\}, \quad Y_1 = \{01, 11, 21, -1 - 1\}, \quad Y_2 = \{02, 12, 22\}.$$ 

One may ask whether or not $m_1$ and $m_2$ of this example should be called semantically independent. Notice that the two states $(1, 1)$ and $(-1, -1)$ (which are in the intersection of a $\rho_1$-equivalence class and a $\rho_2$-equivalence class, thus violating
Definition 2.5(ii)) are just duplicates of each other. More precisely, they are equivalent in the sense that they have exactly the same connections to the other states:

Definition 4.1. Let $S$ be a set and $m_1, m_2 \subseteq S \times S$ two relations on $S$; let $s, t \in S$. Then $s$ and $t$ will be called equivalent with respect to $m_1$ and $m_2$ iff for all $s' \in S$:

$$
(s', s) \in m_1 \iff (s', t) \in m_1,
$$

$$
(s, s') \in m_1 \iff (t, s') \in m_1,
$$

$$
(s', s) \in m_2 \iff (s', t) \in m_2,
$$

$$
(s, s') \in m_2 \iff (t, s') \in m_2.
$$

(Clearly, this relation is indeed an equivalence.)

Two equivalent states can be identified and collapsed into a single state. If this is done to the states $(1, 1)$ and $(-1, -1)$ of Fig. 11, then a smaller state graph is obtained in which $m_1$ and $m_2$ are semantically independent according to (the full) Definition 2.6. The next theorem implies that such identifications are possible whenever Definition 2.5(ii) is the only one violated.

Theorem 4.2. Let $A$ be a set, $m_1, m_2 \subseteq S \times S$ and $\rho_1, \rho_2$ two partitionings of $S$ which satisfy properties 2.5(i), 2.6(ii) and 2.6(iii). Let $X \in \rho_1$ and $Y \in \rho_2$. Then for all $s, t \in X \cap Y$: $s$ is equivalent to $t$ with respect to $m_1, m_2$.

Proof. We prove only that for all $s' \in S$:

$$
(s', s) \in m_1 \Rightarrow (s', t) \in m_1,
$$

and

$$
(s, s') \in m_1 \Rightarrow (t, s') \in m_1.
$$

The other six implications of Definition 4.1 can be proved symmetrically.

Let $s, t \in X \cap Y$ and $s' \in S$.

To prove $(s', s) \in m_1 \Rightarrow (s', t) \in m_1$, let $(s', s) \in m_1$. We know $(s, t) \in \rho_2$ since $s \in Y, t \in Y$ and $Y$ is a $\rho_2$-equivalence class. Also $(s', s) \in m_1$ implies, by property 2.6(ii.a), $(s', s) \in \rho_2$. By the transitivity of $\rho_2$, $(s', t) \in \rho_2$. Furthermore, $(s', s') \in \rho_1$ and $(s', s) \in m_1$ and $(s, t) \in \rho_1$ (since $s \in X, t \in X$ and $X$ is a $\rho_1$-equivalence class); hence $(s', t) \in (\rho_1 \circ m_1 \circ \rho_1)$. By Definition 2.6(iii.a), it follows that $(s', t) \in m_1$.

To prove $(s, s') \in m_1 \Rightarrow (t, s') \in m_1$, let $(s, s') \in m_1$. We have $(t, s) \in \rho_2$ since $t \in Y, s \in Y$ and $Y$ is a $\rho_2$-equivalence class. Also, $(s, s') \in m_1$ implies, by Definition 2.6(ii.a), $(s, s') \in \rho_2$. By the transitivity of $\rho_2$, $(t, s') \in \rho_2$. Furthermore, $(t, s) \in \rho_1$ (since $t \in X$,
Thus, even though property 2.5(ii) is not implied by the rest of Definition 2.6, whenever it is violated the states that violate it are equivalent. They can be collapsed to yield a smaller state graph that satisfies all parts of Definition 2.6. Concerning implementation—in the proof of Theorem 3.1, property 2.5(ii) is used only to prove the injectivity of $\beta$. It is easy to see that, if Definition 2.6 is weakened by dropping property 2.5(ii), then Theorem 3.1 remains valid, provided $\beta$ is weakened to be a surjection rather than a bijection; moreover, $\beta$ identifies only equivalent states. Hence, we may conclude that property 2.5(ii) is not an essential part of the definition of semantic independence. It may be dropped altogether; the resulting programs are still commutative and satisfy a slightly generalized notion of disjoint implementation, with $\beta$ being a surjection but not necessarily a bijection.

To conclude this section, we discuss property 2.5(i), i.e.,

$$\forall X \in \rho_1: \forall Y \in \rho_2: |X \cap Y| \geq 1.$$  

We first investigate whether it is necessary for semantic independence. The following example shows that this is the case. On a five-state space $\{0, 1, 2, 3, 4\}$, define partitions

$$\rho_1 = \{(0, 1), (2, 3), (4)\} \quad \text{and} \quad \rho_2 = \{(0), (1, 2), (3, 4)\},$$

Fig. 12. An example that demonstrates the role of the first half of orthogonality.
and relations

\[ m_1 = \{ (0, 0), (1, 1), (2, 1), (3, 4), (4, 4) \} \]

and

\[ m_2 = \{ (0, 0), (1, 0), (2, 3), (3, 3), (4, 4) \} \]

(see Fig. 12). Then \( m_1 \) and \( m_2 \) satisfy all properties of semantic independence with respect to \( \rho_1 \) and \( \rho_2 \), except property 2.5(i). In addition, both \( m_1 \) and \( m_2 \) are functions, i.e., are total and deterministic. But \( m_1 \) and \( m_2 \) are not commutative: e.g., with input state 2, the final state of \( m_1; m_2 \) is 0, while the final state of \( m_2; m_1 \) is 4. The dependence of \( m_1 \) and \( m_2 \) is, in our definition of semantic independence, captured only by (the violation of) property 2.5(i).

This shows that our definition of semantic independence without property 2.5(i) is too weak. However, with property 2.5(i), it may arguably be too strong. Consider the example depicted in Fig. 13. The following interpretation suggests why \( \alpha_1 \) and \( \alpha_2 \) could be called semantically independent. Consider a two-way traffic road on which there is a narrow stretch that must be protected by two signs, one for each direction. Assume that some electrical device prevents that both signs are green at the same time, but that any other combination (including red/red) is allowed. Then

[Diagram of a two-way traffic road with signs and states labeled 01 and 11, connected by transitions labeled \( \rho_1 \) and \( \rho_2 \).]

var \( x, y : \{0, 1\} \)

(with invariant \( x = 1 \lor y = 1 \);

\( \alpha_1 : x := 1 \)

\( \alpha_2 : y := 1 \)

Fig. 13. Intuitively, but not formally, semantically independent
the two actions "turn sign to red" produce exactly the state graph of Fig. 13 yet, intuitively, might be semantically independent.

Our definition of orthogonality implies that, if \(|S|\) is a prime number, one of \(\rho_1\) and \(\rho_2\) must be the discrete and the other must be the trivial partitioning. In the case of Fig. 13, this contradicts requirement 2.6(iii). Therefore, given our definition of semantic independence, the fact that state 00 is excluded is interpreted as a dependence between the two actions.

5. Concluding remarks

Semantic independence is important for the exploitation of concurrency. Two assignments which satisfy the (syntactic) Bernstein condition can clearly be implemented on two processors without mutual interference. Semantic independence is designed to go beyond the syntactic shape of two programs and detect more hidden possibilities of noninterfering implementations. Thus, if two programs are semantically independent then a "clever" implementation exists which achieves their fully concurrent execution.

Our interest in semantic independence stems from two sources. In [4], the need was felt to distinguish so-called "significant dependencies" between events (execution of actions) from "insignificant dependencies" between events, but no formal definition was offered. In a sense, the concept of semantic independence we have defined here captures formally what has been identified informally as "significant event dependency" in [4]. In general, it does not correspond to "information flow" because two actions that overwrite (but do not read) a common variable are semantically dependent, but do not exchange any information about their values. Only the converse can be asserted: if two actions are semantically independent then they may not exchange any information.

Often, when concurrency is considered, assumptions about the hardware are being made. For example, Owicki based her verification method for parallel programs [8] on the assumption that the hardware puts shared memory accesses in sequence (so-called "memory interlock"). She then claimed that, when the basic components of programs (i.e., assignments and tests) satisfy the Bernstein condition, they can safely be executed in parallel. Theorem 2.10 corroborates her informal claim. The second source of our interest in semantic independence was a desire to eliminate assumptions about the hardware when considering parallelism in programs. By Theorem 3.1, Owicki's assumption of memory interlock can be avoided. We also wanted to be more formal; see [6] for a previous, informal attempt of defining independence without an appeal to hardware properties.

In [9], Pnueli defines the "virtual coarsening" of a set of atomic actions to be their fusion into a bigger action, provided at most one of them contains a critical reference. The latter is a syntactic concept. Virtual coarsening is advantageous
because larger actions of a program usually imply less proof obligations. Using the results of the present paper, this technique may be refined into allowing two atomic actions to be merged provided one of them is semantically independent of the environment. This would replace the syntactic concept of critical reference by our semantic concept, generally yielding larger sets of fusible actions.

In the theory of Petri nets, a definition of independence is known which differs from the one we give here. However, it is shown in [3] that two semantically independent programs can be translated into a Petri net in such a way that the transitions of the net that correspond to the two programs are mutually independent in the sense of net theory.

Although, to our knowledge, no other definition of semantic independence known in the literature is as weak as ours, we have put forward arguments that the definition is not too weak, i.e., that counterexamples could not be found. However, the discussion in Section 5 shows that there is still room for generalizations, i.e., that there are examples which are (arguably) semantically independent but do not satisfy our definition. Definition 2.5(ii) can be omitted to capture some of these cases; a weaker form of disjoint implementability then still holds. To capture other such cases, it seems that property 2.5(i) can be weakened. However, neither did we find an easy way of achieving this, nor did we feel the compelling need for such a generalisation. Hence we are content to leave the question open for future research.

An important topic for future research is to find out how Theorem 3.1 could be given a more syntactic, proof-oriented form. One would have to study the question if there are any classes of relations, situation properly between the semantic independent ones and those satisfying the Bernstein condition, which make Theorem 3.1 constructive, i.e., for which the relations \( \mu \) and \( \nu \) defined there can be constructed systematically. Since these questions do not appear to have an immediate answer, we only raise, but do not attempt to answer, them here.

Theorem 3.1 implies that, in principle, there is no need for shared-read architectures: whenever the program suggests a shared read, a transformation exists that makes it unnecessary. The variable transformations which we have used as representations of the bijection \( \beta \) are reminiscent of coordinate transformations in mathematics and appear to be useful in order to find "clever" implementations. The question may arise why we did not try to formalize such transformations directly. The answer is that it is not always obvious how to express \( \beta \) in terms of transformations.

Another question that may arise concerns the decidability of Definition 2.6: given two arbitrary assignments, is it decidable whether or not they are semantically independent? If infinite value sets are involved then the answer to this question is probably negative. However, we deem the decidability to be of secondary importance compared to the question of having a proper definition of semantic independence to start with and to work with. Also, there are plenty of properties which are theoretically undecidable but of great practical importance; "correctness" is a case in point.
Acknowledgement

We gratefully acknowledge comments by Manfred Broy and Rick Hehner. The example discussed in Fig. 13 is condensed from a seven-state example with similar properties proposed by Ulrich Grude. The scrutiny of Chua-Huang Huang led to an improvement in the proof of Theorem 2.7. We would also like to thank Frau Elisabeth Münch for preparing the figures.

References