Synthesis of Strategies and the Hoare Logic of Angelic Nondeterminism

Konstantinos Mamouras

Cornell University*
mamouras@cs.cornell.edu

Abstract. We study a propositional variant of Hoare logic that can be used for reasoning about programs that exhibit both angelic and demonic nondeterminism. We work in an uninterpreted setting, where the meaning of the atomic actions is specified axiomatically using hypotheses of a certain form. Our logical formalism is entirely compositional and it subsumes the non-compositional formalism of safety games on finite graphs. We present sound and complete Hoare-style (partial-correctness) calculi that are useful for establishing Hoare assertions, as well as for synthesizing implementations. The computational complexity of the Hoare theory of dual nondeterminism is investigated using operational models, and it is shown that the theory is complete for exponential time.

1 Introduction

One source of demonic nondeterminism in a program is its interaction with the environment (e.g., user input, thread scheduling, etc.), which is not under the control of the program. Even in the absence of such "real" nondeterminacy, we may use demonic nondeterminism to represent abstraction and partial knowledge of the state of a computation. Angelic nondeterminism, on the other hand, is used to express nondeterminacy that is under the control of the program. For example, we use angelic nondeterminism when implementation details are left underspecified, but we control how they can be resolved in order to achieve the desired result. The process of resolving these implementation details amounts to *synthesizing* a fully specified program. The term *dual nondeterminism* is used to refer to the combination of angelic and demonic nondeterminism.

In order to reason about dual nondeterminism, one first needs to have a semantic model of how programs with angelic and demonic choices compute. One semantic model that has been used extensively uses a class of mathematical objects that are called monotonic predicate transformers [1] (based on Dijkstra's predicate transformer semantics [4,11]). An equivalent denotational model that is based on binary relations was introduced in [13] (up-closed multirelations) and further investigated in [10]. These relations have an intuitive interpretation as two-round games between the angel and the demon.

* Part of the present work was done while visiting Radboud University Nijmegen.

We are interested here in verifying properties of programs that can be expressed as Hoare assertions [5], that is, formulas of the form $\{p\}f\{q\}$, where f is the program text and p, q denote predicates on the state space, called precondition and postcondition respectively. The formula $\{p\}f\{q\}$ asserts, informally, that starting from any state satisfying the precondition p, the angel has a strategy so that whatever the demon does, the final state of the computation of f (assuming termination) satisfies the postcondition q. This describes a notion of partial correctness, because in the case of divergence (non-termination) the angel wins vacuously. Our language for programs and preconditions/postconditions involves abstract test symbols $p, q, r, \ldots$ and abstract action symbols $a, b, \ldots$ with no fixed interpretation. We constrain their meaning with extra hypotheses: we consider a finite set Φ of Boolean axioms for the tests, and a finite set Ψ of axioms of the form $\{p\}a\{q\}$ for the action letters. So, we typically assert implications of the form $\Phi, \Psi \Rightarrow \{p\}f\{q\}$, which we call *simple Hoare implications*. We want to design a formal system that allows the derivation of the valid Hoare implications. One important desideratum for such a formal system is to also provide us with program text that corresponds to the winning strategy of the angel. Then, the system can be used for the deductive synthesis of programs that satisfy their Hoare specifications.

There has been previous work on deductive methods to reduce angelic nondeterminism and synthesize winning strategies for the angel. The work [2], which is based on ideas of the refinement calculus [1,11], explores a total-correctness Hoare-style calculus to reason about angelic nondeterminism. The analysis is in the first-order interpreted setting, and no completeness or relative completeness results are discussed.

Of particular relevance is the line of work that concerns two-player infinite games played on finite graphs [14]. Such games are useful for analyzing (nonterminating) reactive programs. One of the players represents the "environment", and the other player is the "controller". Computing the strategies that witness the winning regions of the two players amounts to synthesizing an appropriate implementation for the controller. The formalism of games on finite graphs is very convenient for developing an algorithmic theory of synthesis. However, the formalism is non-succinct and, additionally, it is inherently non-compositional. An important class of properties for these games are the so called *safety* properties, which assert that something bad never happens. For such properties, we see that a fully compositional formalism involving usual (terminating) programs and partial-correctness properties suffices.

Our Contribution. We consider a propositionally abstracted language for programs with demonic and angelic choices. Our results are the following:

- We present a sound and *unconditionally* complete calculus for the weak Hoare theory of dual nondeterminism (over the class of all interpretations). We also consider a restricted class of interpretations, where the atomic actions are non-angelic, and we extend our calculus so that it is complete for the Hoare theory of this smaller class (called strong Hoare theory). The proofs of these results rely on the construction of free models.

– We show that (for the free models) the denotational semantics is equivalent to the intended operational semantics. Using this result, we prove that the strong Hoare theory of dual nondeterminism is EXPTIME-complete.
– We consider an extension of our Hoare-style calculus with annotations that denote the winning strategies of the angel. We thus obtain a sound and complete deductive system for the synthesis of angelic strategies.
– Our formalism is shown to subsume that of safety games on finite graphs, hence it provides a compositional method for reasoning about safety in reactive systems. The language of dually nondeterministic program schemes is exponentially more succinct than explicitly represented game graphs, and it is arguably a more natural language for describing algorithms and protocols.

Due to lack of space all proofs will be given in a full version of the paper [9].

2 Preliminaries

In this section we give some preliminary definitions regarding while program schemes with the additional construct $\sqcap$ of demonic nondeterministic choice. First, we present the syntax of these abstract while programs. Then, we give the standard denotational semantics for them, which is based on binary relations.

We consider a two-sorted algebraic language. There is the sort of *tests* and the sort of *programs*. The tests are built up from *atomic tests* and the constants true and false, using the usual Boolean operations: $\neg$ (negation), $\wedge$ (conjunction), and $\vee$ (disjunction). We use the letters $p, q, r, \ldots$ to range over arbitrary tests.

The base programs are the *atomic programs* $a, b, c, \ldots$ (also called *atomic actions*), as well as the constants id (*skip*) and $\bot$ (*diverge*). The programs are constructed using the operations ; (*sequential composition*), if (*conditional*), while (*iteration*), and $\sqcap$ (*demonic nondeterministic choice*). We write $f, g, h, \ldots$ to range over arbitrary programs. So, the programs are given by the grammar:

$$f, g ::= \text{actions } a, b, \ldots \mid \mathsf{id} \mid \bot \mid f; g \mid \mathsf{if}\, p\, \mathsf{then}\, f\, \mathsf{else}\, g \mid \mathsf{while}\, p\, \mathsf{do}\, f \mid f \sqcap g.$$

We also write $p[f, g]$ instead of $\mathsf{if}\, p\, \mathsf{then}\, f\, \mathsf{else}\, g$, and $\mathsf{w}pf$ instead of $\mathsf{while}\, p\, \mathsf{do}\, f$.

We will present the standard denotational semantics of nondeterministic while schemes. Every test is interpreted as a unary predicate on the state space, and every program is interpreted as a binary relation on the state space.

Definition 1 (nondeterministic functions & operations). For a set A, we write $\wp A$ for the *powerset* of A. For sets A and B, we say that a function of type $\phi : A \to \wp B$ is a *nondeterministic function* from A to B. We write $\phi : a \mapsto b$ to mean that $b \in \phi(a)$. We think informally that such a function describes only one kind of nondeterminism (for our purposes here, demonic nondeterminism).

The operations of *(Kleisli) composition* $\mathbf{;}$, *conditional* $(-)\big[-,-\big]$, *binary (nondeterministic) choice* $+$, *arbitrary choice* $\sum$, *identity* 1, *zero* 0, and *iteration* $(\mathbf{wh}\ -\ \mathbf{do}\ -)$ are defined as follows:

$$\phi \mathbf{;} \psi \triangleq \lambda x \in A.\ \textstyle\bigcup_{y \in \phi(x)} \psi(y) : A \to \wp C, \text{ for } \phi : A \to \wp B,\ \psi : B \to \wp C$$

$$P\big[\phi, \psi\big] \triangleq \big(\phi \cap (P \times \wp B)\big) \cup \big(\psi \cap (\sim P \times \wp B)\big), \text{ for } \phi, \psi : A \to \wp B,\ P \subseteq A$$

$$\phi + \psi \triangleq \lambda x \in A.\ \phi(x) \cup \psi(y) : A \to \wp B, \text{ where } \phi, \psi : A \to \wp B$$
$$\textstyle\sum_i \phi_i \triangleq \lambda x \in A.\ \bigcup_i \phi_i(x) : A \to \wp B, \text{ where } \phi_i : A \to \wp B$$
$$1_A \triangleq \lambda x \in A.\ \{x\} : A \to \wp A \quad \text{and} \quad 0_{AB} \triangleq \lambda x \in A.\ \emptyset : A \to \wp B$$
$$\mathbf{wh}\, P\, \mathbf{do}\, \phi \triangleq \textstyle\sum_{n \geq 0} W_n : A \to \wp A, \text{ where } \phi : A \to \wp A \text{ and } P \subseteq A$$
$$W_0 \triangleq P\big[0_{AA}, 1_A\big] \quad \text{and} \quad W_{n+1} \triangleq P\big[\phi; W_n, 1_A\big]$$

where $\sim P = A \setminus P$ above denotes the complement of P w.r.t. A. From the definition of the conditional, we see that $P\big[\phi, \psi\big](x)$ is equal to $\phi(x)$ when $x \in P$, and equal to $\psi(x)$ when $x \notin P$.

Definition 2 (nondeterministic interpretation). An interpretation of the language of nondeterministic while program schemes consists of a nonempty set S, called the *state space*, and an *interpretation function* R. For a program term f, its *interpretation* $R(f) : S \to \wp S$ is a nondeterministic function on S.

The interpretation $R(p)$ of a test p is a unary predicate on S, i.e., $R(p) \subseteq S$. R specifies the meaning of every atomic test, and it extends as follows:

$$R(\mathsf{true}) = S \qquad R(\neg p) = \sim R(p) \qquad R(p \wedge q) = R(p) \cap R(q)$$
$$R(\mathsf{false}) = \emptyset \qquad\qquad\quad R(p \vee q) = R(p) \cup R(q)$$

where $\sim$ is the operation of complementation w.r.t. S, that is, $\sim A = S \setminus A$. Moreover, the interpretation function R specifies the meaning $R(a) : S \to \wp S$ of every atomic program. We extend the interpretation to all program terms:

$$R(\mathsf{id}) = 1_S \qquad R(f;g) = R(f); R(g) \qquad R(p[f,g]) = R(p)\big[R(f), R(g)\big]$$
$$R(\bot) = 0_{SS} \qquad R(f \sqcap g) = R(f) + R(g) \qquad R(\mathsf{w}pf) = \mathbf{wh}\, R(p)\, \mathbf{do}\, R(f)$$

Our definition agrees with the standard relational semantics of while schemes.

3 Angelic and Demonic Nondeterminism

We extend the syntax of nondeterministic while program schemes with the additional construct $\sqcup$ of *angelic (nondeterministic) choice*. So, the grammar for the program terms now becomes:

$$f, g ::= \text{actions } a, b, \ldots \mid \mathsf{id} \mid \bot \mid f; g \mid p[f, g] \mid \mathsf{w}pf \mid f \sqcap g \mid f \sqcup g.$$

We call these program terms *while game schemes*, because they can be considered to be descriptions of games between the angel (who controls the angelic choices) and the demon (who controls the demonic choices). Informally, the angel tries to satisfy the specification, while the demon attempts to falsify it.

We present a relational denotational semantics for while game schemes with abstract atomic actions. A nonempty set S represents the abstract state space, and every test is interpreted as a unary predicate on the state space. Every program term is interpreted as a binary relation from S to $\wp S$.

Consider such a binary relation $f \subseteq S \times \wp S$, which should be thought of as the extension of a game program scheme. Informally, the pair (u, X) is supposed to belong to f when the following holds: if the program starts at state u, then the

angel has a strategy so that whatever the demon does, the final state (supposing that the program terminates) satisfies the predicate X.

The binary relation $f \subseteq S \times \wp S$ encodes both the choices of the angel and the demon, and it can be understood as a two-round game. The angel moves first, and then the demon makes the final move. The options that are available to the angel are given by multiple pairs (u, X_1), (u, X_2), and so on. So, when the game starts at state u, the angel first chooses either X_1, or X_2, or any of the other available options. Suppose that the angel first chooses X_i, where (u, X_i) is in f. Then, during the second round, the demon chooses some final state $v \in X_i$.

When (u, X) is in f, we understand this as meaning that that the angel can guarantee the predicate X when we start at u. So, we should expect that the angel also guarantees any predicate that is weaker than X.

Definition 3 (game functions). For nonempty sets A and B, we say that $f \subseteq A \times \wp B$ is a *game function* from A to B, denoted $f : A \rightsquigarrow B$, if it satifies:

1. The set f is *closed upwards*: $(u, X) \in f$ and $X \subseteq Y \subseteq B \implies (u, Y) \in f$.
2. For every $u \in A$ there is some $X \subseteq B$ with $(u, X) \in f$.

Given Condition (1), we can equivalently require that $(u, B) \in f$ for every $u \in A$, instead of having Condition (2).

Let $f : A \rightsquigarrow B$ be a game function. The *options* of the angel at $u \in A$, which we denote by $f(u)$, is the set $f(u) := \{X \subseteq B \mid (u, X) \in f\}$. In other words, $f(u)$ is the set of all predicates that the angel can guarantee from u.

We say that a game function $f : A \rightsquigarrow B$ is *non-angelic* if for every $u \in A$ there is some $X \subseteq B$ so that $f(u) = \{Y \subseteq B \mid X \subseteq Y\}$. It is easy to see that this $X \subseteq B$ is unique, because the equality $\{Y \subseteq B \mid X_1 \subseteq Y\} = \{Y \subseteq B \mid X_2 \subseteq Y\}$ implies that $X_1 = X_2$. Essentially, the definition says that the angel always has exactly one minimal choice: for every $u \in A$ there is exactly one minimal predicate X that the angel can guarantee.

Definition 4 (lifting & non-angelic game functions). When $f : A \rightsquigarrow B$ is a non-angelic game function, there is essentially only demonic nondeterminism. So, the same information can be provided by a nondeterministic function $A \to \wp B$. Indeed, we see easily that $f : A \rightsquigarrow B$ is non-angelic iff there exists some function $\phi : A \to \wp B$ so that $f = \mathsf{lift}\,\phi$, where

$$\mathsf{lift}\,\phi \triangleq \{(u, Y) \mid u \in A,\ \phi(u) \subseteq Y\} : A \rightsquigarrow B$$

defines the *lifting operation* lift. The definition says that for every $u \in A$ and $Y \subseteq B$: $(u, Y) \in \mathsf{lift}\,\phi$ iff $\phi(u) \subseteq Y$.

Definition 5 (operations on game functions). We define a binary *composition* operation for game functions, whose typing rule and definition are:

$$\frac{f : A \rightsquigarrow B \quad g : B \rightsquigarrow C}{f ; g : A \rightsquigarrow C} \qquad (u, Z) \in (f ; g) \Leftrightarrow \text{there is } Y \subseteq B \text{ s.t. } (u, Y) \in f, \text{ and } (v, Z) \in g \text{ for every } v \in Y.$$

The *(semantic) conditional operation* is given as follows:

$$P[f, g] \triangleq \big(f \cap (P \times \wp B)\big) \cup \big(g \cap (\sim P \times \wp B)\big), \text{ for } f, g : A \rightsquigarrow B \text{ and } P \subseteq A,$$

where $\sim P = A \setminus P$ is the complement of P w.r.t. A. The *angelic choice* operation $\sqcup$ for game functions is defined by:

$$f \sqcup g \triangleq f \cup g, \text{ where } f, g : A \rightsquigarrow B.$$

As expected, the angelic choice operation increases the options available to the angel. Now, we define the *demonic choice* operation $\sqcap$ for game functions as:

$$f \sqcap g \triangleq \{(u, X \cup Y) \mid (u, X) \in f,\ (u, Y) \in g\}, \text{ where } f, g : A \rightsquigarrow B.$$

So, demonic choice increases the options of the demon. The above definition is equivalent to $f \sqcap g = f \cap g$. The *identity function* $\mathbb{1}_A : A \rightsquigarrow A$ is defined by

$$\mathbb{1}_A \triangleq \{(u, X) \mid u \in A \text{ and } u \in X\}.$$

So, $\mathbb{1}_A$ is the smallest game function that contains $(u, \{u\})$ for every $u \in A$. Informally, this definition says that on input u, the angel guarantees output u in the identity game. The *diverging* game function $\mathbb{0}_{AB} : A \rightsquigarrow B$ is given by

$$\mathbb{0}_{AB} \triangleq \{(u, X) \mid u \in A \text{ and } X \subseteq B\} = A \times \wp B.$$

The intuition for the definition of $\mathbb{0}_{AB}$ is that when the program diverges, the demon cannot lead the game to an error state, therefore the angel can guarantee anything. This describes a notion of partial correctness. Finally, the *(semantic) while operation* ($\mathbf{wh}\ -\ \mathbf{do}\ -$) has the following typing rule and definition:

$$\frac{P \subseteq A \qquad f : A \rightsquigarrow A}{\mathbf{wh}\, P\, \mathbf{do}\, f \triangleq \bigcap_{\kappa \in \mathbf{Ord}} W_\kappa : A \rightsquigarrow A} \qquad \begin{aligned} W_0 &= P\big[\mathbb{0}_{AA}, \mathbb{1}_A\big] \\ W_{\kappa+1} &= P\big[f; W_\kappa, \mathbb{1}_A\big] \\ W_\lambda &= \textstyle\bigcap_{\kappa<\lambda} W_\kappa, \text{ limit ordinal } \lambda \end{aligned}$$

The sets $W_0 \supseteq W_1 \supseteq W_2 \supseteq \cdots \supseteq W_\kappa \supseteq \cdots$ form a decreasing chain. That is, $\kappa \leq \lambda$ implies $W_\kappa \supseteq W_\lambda$, for any ordinals κ and λ.

We note that the above definition gives the while operation as a greatest fixpoint. This is not surprising, because the semantics we consider is meant to be useful for reasoning about *safety properties*. As we will see, this definition agrees with the standard least fixpoint definition of while loops when there is only one kind of nondeterminism (Lemma 6). More importantly, we will prove that our definition is *exactly right*, becauses it agrees with the intended operational semantics of dual nondeterminism (Proposition 28).

Lemma 6 (lift commutes with the operations). Let ϕ and ψ be nondeterministic functions, and P be a predicate. Then, the following hold:

$$\mathsf{lift}\, 0_{AB} = \mathbb{0}_{AB} \quad \mathsf{lift}(\phi; \psi) = (\mathsf{lift}\, \phi); (\mathsf{lift}\, \psi) \quad \mathsf{lift}(P\big[\phi, \psi\big]) = P\big[\,\mathsf{lift}\, \phi, \mathsf{lift}\, \psi\big]$$
$$\mathsf{lift}\, 1_A = \mathbb{1}_A \quad \mathsf{lift}(\phi + \psi) = (\mathsf{lift}\, \phi) \sqcap (\mathsf{lift}\, \psi) \quad \mathsf{lift}(\mathbf{wh}\, P\, \mathbf{do}\, \phi) = \mathbf{wh}\, P\, \mathbf{do}\, (\mathsf{lift}\, \phi)$$

Essentially, the lemma says that the game function operations are a generalization of the nondeterministic function operations.

For a nondeterministic function $\phi : A \to \wp B$ and a game function $f : A \rightsquigarrow B$, we say that ϕ *implements* f if $\mathsf{lift}\, \phi \subseteq f$. So, ϕ implements f when it resolves (in some possible way) the angelic nondeterminism of f.

Definition 7 (game interpretation of programs). As in the case of nondeterministic program schemes (Definition 2), an interpretation of the language

of while game schemes consists of a nonempty *state space* S and an *interpretation function* I. For a program term f, its *interpretation* $I(f) : S \rightsquigarrow S$ is a game function on S. The function I specifies the meaning of every atomic test, and extends to all tests in the obvious way. Moreover, I specifies the meaning $I(a) : S \rightsquigarrow S$ of every atomic action. It extends as: $I(\mathsf{id}) = \mathbb{1}_S$, $I(\bot) = \mathbb{0}_{SS}$, and

$$I(f;g) = I(f);I(g) \quad I(f \sqcup g) = I(f) \sqcup I(g) \quad I(p[f,g]) = I(p)\big[I(f), I(g)\big]$$
$$I(f \sqcap g) = I(f) \sqcap I(g) \qquad I(\mathsf{w}pf) = \mathbf{wh}\, I(p)\, \mathbf{do}\, I(f)$$

We say that the game interpretation I *lifts* the nondeterministic interpretation R if they have the same state space, and additionally: (i) $I(p) = R(p)$ for every atomic test p, and (ii) $I(a) = \mathsf{lift}\, R(a)$ for every atomic program a. We also say that I *is the lifting of* R.

4 Hoare Formulas and their Meaning

In this section, we present formulas that are used to specify programs. The basic formulas are called Hoare assertions, and we also consider assertions under certain hypotheses of a simple form (Hoare implications).

Definition 8 (tests and entailment). Let I be an interpretation of tests. For a test p and a state $u \in S$, we write $I, u \models p$ when $u \in I(p)$. We read this as: "the state u satisfies p (under I)". When $I, u \models p$ for every state $u \in S$, we say that I *satisfies* p, and we write $I \models p$. For a set Φ of tests, the interpretation I *satisfies* Φ if it satisfies every test in Φ. We then write $I \models \Phi$. Finally, we say that Φ *entails* p, denoted $\Phi \models p$, if $I \models \Phi$ implies $I \models p$ for every I.

Definition 9 (Hoare assertions). An expression $\{p\}f\{q\}$, where p and q are tests and f is a program term, is called a *Hoare assertion*. The test p is called the *precondition* and the test q is called the *postcondition* of the assertion. Informally, the formula $\{p\}f\{q\}$ says that when the program f starts at a state satisfying the predicate p, then the angel has a strategy so that whatever the demon does, the final state (upon termination) satisfies the predicate q. The Hoare assertion $\{p\}a\{q\}$, where a is an atomic program, is called a *simple Hoare assertion*. More formally, consider an interpretation I. We say that I *satisfies* $\{p\}f\{q\}$, and we write $I \models \{p\}f\{q\}$, when the following holds for every state $u \in S$: $I, u \models p$ implies that $(u, I(q)) \in I(f)$.

Definition 10 (simple Hoare implications). Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. We call the expression

$$\Phi, \Psi \Rightarrow \{p\}f\{q\}$$

a *simple Hoare implication*. The tests in Φ and the simple assertions in Ψ are the *hypotheses* of the implication, and the Hoare assertion $\{p\}f\{q\}$ is the *conclusion*.

Let I be an interpretation of tests and actions. We say that I *satisfies* the implication $\Phi, \Psi \Rightarrow \{p\}f\{q\}$, which we denote by $I \models \Phi, \Psi \Rightarrow \{p\}f\{q\}$, when the following holds: If the interpretation I satisfies every test in Φ and every assertion in Ψ, then I satisfies the assertion $\{p\}f\{q\}$. An implication $\Phi, \Psi \Rightarrow \{p\}f\{q\}$ is

valid, denoted $\Phi, \Psi \models \{p\}f\{q\}$, if every interpretation satisfies it. The set of all valid Hoare implications forms the *weak Hoare theory* of while game schemes.

Definition 11 (Boolean Atoms & Φ-consistency). Suppose that we have fixed a finite set of atomic tests. For an atomic test p, the expressions p and $\neg p$ are called *literals* for p (*positive* and *negative* respectively). Fix an enumeration $p_1, p_2, \ldots, p_k$ of the atomic tests. A *Boolean atom* (or simply *atom*) is an expression $\ell_1 \ell_2 \cdots \ell_k$, where every ℓ_i is a literal for p_i. We use lowercase letters $\alpha, \beta, \gamma, \ldots$ from the beginning of the Greek alphabet to range over atoms. An atom is essentially a conjunction of literals, and it can also be thought of as a propositional truth assignment. We write $\alpha \leq p$ to mean that the atom α satisfies the test p. We denote by At the set of all atoms.

Assume that Φ is a finite set of tests. We say that an atom α is *Φ-consistent* if $\alpha \leq p$ for every test p in Φ. We write At_Φ for the set of all Φ-consistent atoms.

Definition 12 (the free test interpretation). Let Φ be a finite set of tests. We define the interpretation I_Φ on tests, which is called the *free test interpretation* w.r.t. Φ. The state space is the set At_Φ of Φ-consistent atoms, and every test is interpreted as a unary predicate on At_Φ. For an atomic test p, define $I_\Phi(p) := \{\alpha \in \mathsf{At}_\Phi \mid \alpha \leq p\}$ to be the set of Φ-consistent atoms that satisfy p.

An easy induction on the structure of tests proves that for every (atomic or composite) test p, $I_\Phi(p)$ is equal to the set of Φ-consistent atoms that satisfy p.

Note 13 (complete Boolean calculus). We assume that we have a complete Boolean calculus, with which we derive judgments $\Phi \vdash p$, where Φ is a finite set of tests and p is a test. This means that the statements $\Phi \models p$, $I_\Phi \models p$, $I_\Phi(p) = \mathsf{At}_\Phi$, and $\Phi \vdash p$ are all equivalent. Moreover, $I_\Phi(p) \subseteq I_\Phi(q)$ iff $\Phi \vdash p \to q$.

5 A Hoare Calculus for While Game Schemes

In this section we propose a Hoare-style calculus (Table 1), which is used for deriving simple Hoare implications that involve while game schemes. As we will show, the calculus of Table 1 is sound and complete for the weak Hoare theory of while game schemes. Establishing soundness is a relatively straightforward result. The most interesting part is the soundness of the (loop) rule for while loops. The observation is that the loop invariant defines a "safe region" of the game, and the angel has a strategy to keep a play within this region.

Theorem 14 (soundness). The Hoare calculus of Table 1 is sound.

5.1 First Completeness Theorem: Weak Hoare Theory

We will now prove the completeness of the Hoare calculus of Table 1 with respect to the class of all interpretations. This means that we consider arbitrary interpretations of the atomic programs $a, b, \ldots$ as game functions. So, the deductive system of Table 1 is complete for the weak Hoare theory of while game schemes. Note that this is an *unconditional* completeness result (no extra assumptions), not a relative completeness theorem [3].

$$\frac{\{p\}a\{q\} \text{ in } \Psi}{\Phi,\Psi \vdash \{p\}a\{q\}}\ (\mathsf{hyp}) \qquad \frac{}{\Phi,\Psi \vdash \{p\}\mathsf{id}\{p\}}\ (\mathsf{skip}) \qquad \frac{}{\Phi,\Psi \vdash \{p\}\bot\{q\}}\ (\mathsf{dvrg})$$

$$\frac{\Phi,\Psi \vdash \{p\}f\{q\} \quad \Phi,\Psi \vdash \{q\}g\{r\}}{\Phi,\Psi \vdash \{p\}f;g\{r\}}\ (\mathsf{seq}) \qquad \frac{\Phi,\Psi \vdash \{q \wedge p\}f\{r\} \quad \Phi,\Psi \vdash \{q \wedge \neg p\}g\{r\}}{\Phi,\Psi \vdash \{q\}\mathsf{if}\, p\, \mathsf{then}\, f\, \mathsf{else}\, g\{r\}}\ (\mathsf{cond})$$

$$\frac{\Phi,\Psi \vdash \{r \wedge p\}f\{r\}}{\Phi,\Psi \vdash \{r\}\mathsf{while}\, p\, \mathsf{do}\, f\{r \wedge \neg p\}}\ (\mathsf{loop})$$

$$\frac{\Phi,\Psi \vdash \{p\}f_i\{q\}}{\Phi,\Psi \vdash \{p\}f_1 \sqcup f_2\{q\}}\ (\mathsf{ang}_i) \qquad \frac{\Phi,\Psi \vdash \{p\}f\{q\} \quad \Phi,\Psi \vdash \{p\}g\{q\}}{\Phi,\Psi \vdash \{p\}f \sqcap g\{q\}}\ (\mathsf{dem})$$

$$\frac{\Phi \vdash p' \to p \qquad \Phi,\Psi \vdash \{p\}f\{q\} \qquad \Phi \vdash q \to q'}{\Phi,\Psi \vdash \{p'\}f\{q'\}}\ (\mathsf{weak})$$

$$\frac{\Phi,\Psi \vdash \{p_1\}f\{q\} \quad \Phi,\Psi \vdash \{p_2\}f\{q\}}{\Phi,\Psi \vdash \{p_1 \vee p_2\}f\{q\}}\ (\mathsf{join}) \qquad \begin{array}{ll} \Phi,\Psi \vdash \{\mathsf{false}\}f\{q\} & (\mathsf{join}_0) \\ \Phi,\Psi \vdash \{p\}f\{\mathsf{true}\} & (\mathsf{meet}_0) \end{array}$$

Table 1. *Game Hoare Logic*: A sound and complete Hoare-style calculus for while program schemes with angelic and demonic nondeterministic choice.

Definition 15 (the free game interpretation). Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. We define the *free game interpretation* $I_{\Phi\Psi}$ (w.r.t. Φ and Ψ) to have At_Φ as state space, and to interpret the tests as I_Φ (the free test interpretation w.r.t. Φ, see Definition 12) does. Moreover, the interpretation $I_{\Phi\Psi}(a) : \mathsf{At}_\Phi \rightsquigarrow \mathsf{At}_\Phi$ of the atomic action a is given by: for every Φ-consistent atom α,

- $(\alpha, \mathsf{At}_\Phi) \in I_{\Phi\Psi}(a)$, and for every subset $X \subsetneq \mathsf{At}_\Phi$,
- $(\alpha, X) \in I_{\Phi\Psi}(a)$ iff there exists $\{p\}a\{q\} \in \Psi$ s.t. $\alpha \leq p$ and $I_\Phi(q) \subseteq X$.

Lemma 16. Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. The free game interpretation $I_{\Phi\Psi}$ satisfies all formulas in Φ and Ψ.

Theorem 17 (completeness). Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. For every program term f and every Φ-consistent atom α, $(\alpha, X) \in I_{\Phi\Psi}(f)$ implies that $\Phi,\Psi \vdash \{\alpha\}f\{\bigvee X\}$.

Corollary 18 (completeness). Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. For every program f, the following are equivalent:

(1) $\Phi,\Psi \models \{p\}f\{q\}$.
(2) For every Φ-consistent $\alpha \leq p$, the pair $(\alpha, I_\Phi(q))$ is in $I_{\Phi\Psi}(f)$.
(3) $\Phi,\Psi \vdash \{p\}f\{q\}$.

Corollary 18 gives us a decision procedure for the weak Hoare theory of dual nondeterminism. Given a Hoare implication $\Phi,\Psi \Rightarrow \{p\}f\{q\}$, we simply have to compute the free interpretation $I_{\Phi\Psi}(f) \subseteq \mathsf{At}_\Phi \times \wp\mathsf{At}_\Phi$, which is a finite object. Observe that $I_{\Phi\Psi}(f)$ is of doubly exponential size. We will see later that, with some more work, we can devise a faster algorithm of exponential complexity.

$$\frac{\Phi, \Psi \vdash \{p\}a\{q_1\} \qquad \Phi, \Psi \vdash \{p\}a\{q_2\}}{\Phi, \Psi \vdash \{p\}a\{q_1 \wedge q_2\}}\ (a\text{-}\mathsf{meet})$$

Table 2. A rule that is sound when the atomic actions are interpretated as non-angelic game functions. That is, (meet) is sound for the class *Dem*.

5.2 Second Completeness Theorem: Strong Hoare Theory

The completeness theorem of Section 5.1 concerns the theory generated by the class of all interpretations, that is, when the atomic programs are allowed to be interpreted as any game function. However, for most realistic applications the atomic actions $a, b, \ldots$ correspond to computational operations (e.g., variable assignments $x := t$, etc.) that involve no angelic nondeterministic choice. This leads us to consider a strictly smaller class of interpretations, and thus the question is raised of whether this smaller class has the same Hoare theory.

Definition 19 (validity over a class of interpretations). We fix a language with atomic tests and atomic actions. Let $\mathcal{C}$ be a class of interpretations of the atomic symbols (extending to all tests and programs in the usual way). We say that a Hoare implication $\Phi, \Psi \Rightarrow \{p\}f\{q\}$ is *valid in* $\mathcal{C}$ (or *$\mathcal{C}$-valid*) if every interpretation I in $\mathcal{C}$ satisfies the implication. We then write $\Phi, \Psi \models_{\mathcal{C}} \{p\}f\{q\}$. The set of all $\mathcal{C}$-validities is called the *Hoare theory of* $\mathcal{C}$.

Let *All* be the class of all interpretations. Observe that an implication is valid iff it is valid in *All*. Now, let $Dem \subseteq All$ be the strict subclass of interpretations where the atomic actions are interpreted as non-angelic game functions.

Lemma 20 (soundness). The rule (meet) of Table 2, where a is an atomic action, is sound for the class *Dem* of interpretations.

Lemma 20 also establishes that the Hoare theory of *Dem* is different from the Hoare theory of *All*. Strictly more implications hold, when we restrict attention to the interpretations of *Dem*. For example, consider the set of hypotheses Ψ, which consists of the two simple assertions $\{p\}a\{q\}$ and $\{p\}a\{r\}$, where p, q, r are distinct atomic tests. Observe that the implication $\Psi \Rightarrow \{p\}a\{q \wedge r\}$ is valid in *Dem* (by Lemma 20), but it is not valid in *All* (by virtue of Corollary 18).

Definition 21 (the free non-angelic interpretation). Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. For an atomic action a, define the nondeterministic interpretation $R_{\Phi\Psi}(a) : \mathsf{At}_\Phi \to \wp\mathsf{At}_\Phi$ as

$$R_{\Phi\Psi}(a)(\alpha) \triangleq \{\beta \in \mathsf{At}_\Phi \mid \text{for every } \{p\}a\{q\} \in \Psi \text{ with } \alpha \leq p, \text{ we have } \beta \leq q\}.$$

We define the *free non-angelic interpretation* $J_{\Phi\Psi}$ (w.r.t. Φ and Ψ) to have At_Φ as state space, and to interpret the tests as I_Φ (the free test interpretation w.r.t. Φ, see Definition 12) does. Moreover, the interpretation $J_{\Phi\Psi}(a) : \mathsf{At}_\Phi \rightsquigarrow \mathsf{At}_\Phi$ of the atomic action a is given by $J_{\Phi\Psi}(a) := \mathsf{lift}\, R_{\Phi\Psi}(a)$.

Lemma 22. Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. The free non-angelic interpretation $J_{\Phi\Psi}$ satisfies both Φ and Ψ.

Recall that we used the symbol $\vdash$ in Section 5 to denote provability in the Hoare-style system of Table 1. Now, we will use the symbol $\vdash_d$ to denote provability in the Hoare-style system that extends the calculus of Table 1 with the additional rule (meet) shown in Table 2.

Theorem 23 (completeness). Let Φ be a finite set of tests, and Ψ be a finite set of simple Hoare assertions. For every program term f and every Φ-consistent atom α, $(\alpha, Y) \in J_{\Phi\Psi}(f)$ implies that $\Phi, \Psi \vdash_d \{\alpha\} f \{\bigvee Y\}$.

Corollary 24 (completeness). Let Φ and Ψ be finite sets of tests and simple Hoare assertions respectively. For every program f, the following are equivalent:

(1) $\Phi, \Psi \models_{Dem} \{p\} f \{q\}$.
(2) For every Φ-consistent $\alpha \leq p$, the pair $(\alpha, I_\Phi(q))$ is in $J_{\Phi\Psi}(f)$.
(3) $\Phi, \Psi \vdash_d \{p\} f \{q\}$.

The results of this section imply that the Hoare theory of the class *Dem*, which we also call the *strong Hoare theory* of while game schemes, can be reduced to the weak Hoare theory of the class *All*. Let $\Phi, \Psi \Rightarrow \{p\} f \{q\}$ be an arbitrary Hoare implication. W.l.o.g. the axioms in Ψ are of the form $\{\alpha\} a \{q\}$, where α is an atom and a is an atomic action. Now, define Ψ' to be the set of hypotheses that results from Ψ by replacing the axioms $\{\alpha\} a \{q_i\}$ involving α, a by a single axiom $\{\alpha\} a \{\bigwedge_i q_i\}$. The crucial observation is that the interpretation $J_{\Phi\Psi}$ is the same as $I_{\Phi\Psi'}$. Using our two completeness results of Corollary 18 and Corollary 24, it follows that $\Phi, \Psi \vdash_d \{p\} f \{q\}$ iff $\Phi, \Psi' \vdash \{p\} f \{q\}$.

6 Operational Model & Complexity

In this section we investigate the computational complexity of the strong Hoare theory of while game schemes. We prove that this theory is complete for exponential time. In order to obtain the EXPTIME upper bound, we consider a standard operational model that corresponds to the free game interpretation. We establish that our denotational semantics coincides in a precise sense to the operational semantics. The operational model is a safety game on a finite graph, and we can decide validity by computing the winning regions of the players. The lower bound of EXPTIME-hardness is obtained with a reduction from alternating Turing machines with polynomially bounded tapes.

First, we restrict slightly the syntax of program terms by eliminating the diverging $\bot$ program, and by forbidding compositions $(f; g); h$ that associate to the left. These are not really limitations, because $\bot$ is semantically equivalent to the infinite loop $\mathsf{while}\,\mathsf{true}\,\mathsf{do}\,\mathsf{id}$, and $(f; g); h$ is equivalent to $f; (g; h)$. We define the syntactic categories *factor* and *term* with the following grammars:

$$\text{factor } e ::= a \mid \mathsf{id} \mid p[f, g] \mid \mathsf{w}pf \mid f \sqcup g \mid f \sqcap g \qquad \text{terms } f, g ::= e \mid e; f$$

A term according to the above definition is a nonempty list of factors. We write @ for the concatenation of terms: $e@g = e; g$ and $(e; f)@g = e; (f@g)$.

$$(\alpha, a) \to (I_\Phi(q), \mathsf{id}), \text{ where } \{\alpha\}a\{q\} \in \Psi$$
$$(\alpha, a; h) \to (I_\Phi(q), \mathsf{id}; h), \text{ where } \{\alpha\}a\{q\} \in \Psi$$

$(\alpha, \mathsf{id}) \to$	$(\alpha, \mathsf{id}; h) \to (\alpha, h)$
$(\alpha, p[f, g]) \to (\alpha, f)$, if $\alpha \leq p$	$(\alpha, p[f, g]; h) \to (a, f@h)$, if $\alpha \leq p$
$(\alpha, p[f, g]) \to (\alpha, g)$, if $\alpha \leq \neg p$	$(\alpha, p[f, g]; h) \to (a, g@h)$, if $\alpha \leq \neg p$
$(\alpha, \mathsf{w}pf) \to (\alpha, f@\mathsf{w}pf)$, if $\alpha \leq p$	$(\alpha, (\mathsf{w}pf); h) \to (\alpha, f@(\mathsf{w}pf); h)$, if $\alpha \leq p$
$(\alpha, \mathsf{w}pf) \to (\alpha, \mathsf{id})$, if $\alpha \leq \neg p$	$(\alpha, (\mathsf{w}pf); h) \to (\alpha, \mathsf{id}; h)$, if $\alpha \leq \neg p$
$(\alpha, f \sqcup g) \to (\alpha, f),\ (\alpha, g)$	$(\alpha, (f \sqcup g); h) \to (\alpha, f@h),\ (\alpha, g@h)$
$(\alpha, f \sqcap g) \to (\alpha, f),\ (\alpha, g)$	$(\alpha, (f \sqcap g); h) \to (\alpha, f@h),\ (\alpha, g@h)$

$$(X, f) \to (\alpha, f), \text{ where } \alpha \in X \subseteq \mathsf{At}_\Phi$$

Table 3. The operational model that corresponds to the free game interpretation $I_{\Phi\Psi}$.

Definition 25 (closure & the $\to$ relation on terms). We define the *closure* function $C(\cdot)$ that sends a term to a finite set of terms.

$$C(a) = \{a, \mathsf{id}\} \quad C(\mathsf{w}pf) = \{\mathsf{w}pf, \mathsf{id}\} \cup C(f)@\mathsf{w}pf \quad C(e; f) = C(e)@f \cup C(f)$$
$$C(\mathsf{id}) = \{\mathsf{id}\} \quad C(f \oplus g) = \{f \oplus g\} \cup C(f) \cup C(g)$$

where $(- \oplus -)$ is any of the constructors $(- \sqcup -)$, $(- \sqcap -)$, or $p[-, -]$. We define the relation $\to$ on terms as follows:

$$a \to \mathsf{id} \quad \mathsf{w}pf \to f@\mathsf{w}pf, \mathsf{id} \quad f \oplus g \to f, g \quad \mathsf{id}; h \to h$$
$$a; h \to h \quad \mathsf{w}pf; h \to f@(\mathsf{w}pf); h, \mathsf{id}; h \quad (f \oplus g); h \to f@h, g@h$$

We write $\to^*$ for the reflexive transitive closure of $\to$. The definition of $\to$ says, in particular, that id has no successor. The while loop $\mathsf{w}pf$ has exactly two successors, namely $f@\mathsf{w}pf$ and id.

Lemma 26 (closure & reachability). Let f be a program term. The cardinality of $C(f)$ is linear in the size $|f|$ of f, in fact, $|C(f)| \leq 2|f|$. Moreover, $C(f)$ is equal to the set $\{f' \mid f \to^* f'\}$ of terms that are reachable from f via $\to$.

Definition 27 (operational model). Fix a finite set Φ of tests, and a finite set Ψ of simple Hoare assertions. W.l.o.g. we assume that Ψ contains exactly one assertion $\{\alpha\}a\{q\}$ for every atomic program a and every Φ-consistent atom α. Let f be a program term, and $E \subseteq \mathsf{At}_\Phi$ be a set of *error atoms*. We define the *operational model* for Φ, Ψ, f, E, denoted $G_{\Phi\Psi}(f, E)$, to be the safety game

$$G_{\Phi\Psi}(f, E) = (V, V_0, V_1, \to, E \times \{\mathsf{id}\}),$$

where $V = (\mathsf{At}_\Phi \times C(f)) \cup \bigcup_{\{\alpha\}a\{q\} \in \Psi} (I_\Phi(q) \times C(f))$ and the transition relation $\to$ is defined in Table 3.

- The 0-vertices $V_0 \subseteq V$ consist of the pairs of the form $(\alpha, f \sqcup g)$, as well as (α, a) and $(\alpha, a; h)$ for atomic program a.
- The 1-vertices $V_1 \subseteq V$ consist of the pairs $(\alpha, f \sqcap g)$, as well as (X, f) where $X \subseteq \mathsf{At}_\Phi$ is equal to some $I_\Phi(q)$ with $\{\alpha\}a\{q\} \in \Psi$.

The terminal vertices are the pairs (α, id), and the error vertices are $E \times \{\mathsf{id}\}$.

Proposition 28 (operational & denotational semantics). Let Φ be a finite set of tests, Ψ be a finite set of simple Hoare assertions, f be a program term, $\alpha \in \mathsf{At}_\Phi$, and $X \subseteq \mathsf{At}_\Phi$. Then, $(\alpha, X) \in I_{\Phi\Psi}(f)$ iff Player 0 has a winning strategy from the vertex (α, f) in the safety game $G_{\Phi\Psi}(f, {\sim}X)$, where ${\sim}X = \mathsf{At}_\Phi \setminus X$.

Theorem 29 (complexity upper & lower bound). The strong Hoare theory (over the class *Dem*) of while game schemes is EXPTIME-complete.

It is an immediate corollary of the above theorem that the weak Hoare theory (over the class *All*) can also be decided in exponential time.

7 A Complete Hoare-style Calculus for Synthesis

We introduce in Table 4 a Hoare-style calculus which can be used for the deductive synthesis of $\sqcup$-free programs that satisfy a Hoare specification. It is based on the complete calculus for the Hoare theory of the class *Dem*, which contains interpretations assigning non-angelic game functions (Def. 3) to the atomic programs (Table 1 with extra rule of a-meet of Table 2). The main differences are:

(i) The rules join_0 and meet_0 (of Table 1) have been weakened into the rules a-join_0 and a-meet_0 (this is inconsequential).

(ii) Every conclusion $\{p\}f\{q\}$ is decorated with a $\sqcup$-free program term ϕ, which satisfies the specification $\{p\}\phi\{q\}$ and implements a winning strategy for the angel in the safety game described by the assertion $\{p\}f\{q\}$.

Another difference that deserves mention is the introduction in Table 4 of two new variants (join') and (join'') of the rule (join). These rules are not necessary for completeness and they can be omitted without breaking our theorems, but they are useful from a practical viewpoint. The new rules (join') and (join'') are sound, and they allow useful shortcuts in the synthesis of $\sqcup$-free programs.

Theorem 30 (soundness). Suppose that a judgment $\Phi, \Psi \vdash \phi : \{p\}f\{q\}$ is derivable using the Hoare-style calculus of Table 4. The following hold:

1. Every game interpretation I in *Dem* satisfies the formula $\Phi, \Psi \Rightarrow \{p\}f\{q\}$.
2. Every nondeterministic interpretation R satisfies $\Phi, \Psi \Rightarrow \{p\}\phi\{q\}$.
3. Let R be a nondeterministic interpretation, and I be the game interpretation that lifts R (see Definition 7). Then, $\mathsf{lift}\, R(\phi) \subseteq I(f)$.

Part (3) of the theorem says that $R(\phi)$ implements $I(f)$ when I lifts R.

Theorem 31 (completeness). Let Φ and Ψ be finite sets of tests and simple Hoare assertions respectively, and f be a program s.t. $\Phi, \Psi \models_{Dem} \{p\}f\{q\}$. Then, there exists a $\sqcup$-free program ϕ such that $\Phi, \Psi \vdash \phi : \{p\}f\{q\}$.

Finally, we will see that solving safety games on finite graphs can be reduced to deciding the *Dem*-validity of a Hoare implication involving a while game scheme that simulates the safety game. This reduction thus gives us a compositional deductive way of designing winning strategies for safety games. Let $G = (V_0, V_1, R, E)$ be a safety game. For every vertex $u \in V = V_0 \cup V_1$, introduce an atomic test p_u, which asserts that the token is currently on the vertex u. We

$$\frac{\{p\}a\{q\} \text{ in } \Psi}{\Phi,\Psi \vdash a : \{p\}a\{q\}}\ (\mathsf{hyp}) \qquad \frac{}{\Phi,\Psi \vdash \mathsf{id} : \{p\}\mathsf{id}\{p\}}\ (\mathsf{skip}) \qquad \frac{}{\Phi,\Psi \vdash \bot : \{p\}\bot\{q\}}\ (\mathsf{dvrg})$$

$$\frac{\Phi,\Psi \vdash \phi : \{p\}f\{q\} \quad \Phi,\Psi \vdash \psi : \{q\}g\{r\}}{\Phi,\Psi \vdash \phi;\psi : \{p\}f;g\{r\}}\ (\mathsf{seq}) \qquad \frac{\Phi,\Psi \vdash \phi : \{q \wedge p\}f\{r\} \quad \Phi,\Psi \vdash \psi : \{q \wedge \neg p\}g\{r\}}{\Phi,\Psi \vdash p[\phi,\psi] : \{q\}\mathsf{if}\ p\ \mathsf{then}\ f\ \mathsf{else}\ g\{r\}}\ (\mathsf{cond})$$

$$\frac{\Phi,\Psi \vdash \phi : \{r \wedge p\}f\{r\}}{\Phi,\Psi \vdash \mathsf{W}p\phi : \{r\}\mathsf{while}\ p\ \mathsf{do}\ f\{r \wedge \neg p\}}\ (\mathsf{loop})$$

$$\frac{\Phi,\Psi \vdash \phi : \{p\}f_i\{q\}}{\Phi,\Psi \vdash \phi : \{p\}f_1 \sqcup f_2\{q\}}\ (\mathsf{ang}_i) \qquad \frac{\Phi,\Psi \vdash \phi : \{p\}f\{q\} \quad \Phi,\Psi \vdash \psi : \{p\}g\{q\}}{\Phi,\Psi \vdash \phi \sqcap \psi : \{p\}f \sqcap g\{q\}}\ (\mathsf{dem})$$

$$\frac{\Phi \vdash p' \to p \quad \Phi,\Psi \vdash \phi : \{p\}f\{q\} \quad \Phi \vdash q \to q'}{\Phi,\Psi \vdash \phi : \{p'\}f\{q'\}}\ (\mathsf{weak})$$

$$\frac{\Phi,\Psi \vdash \phi_1 : \{p_1\}f\{q\} \quad \Phi,\Psi \vdash \phi_2 : \{p_2\}f\{q\}}{\Phi,\Psi \vdash p_1[\phi_1,\phi_2] : \{p_1 \vee p_2\}f\{q\}}\ (\mathsf{join}) \qquad \frac{}{\Phi,\Psi \vdash a : \{\mathsf{false}\}a\{q\}}\ (a\text{-}\mathsf{join}_0)$$

$$\frac{\Phi,\Psi \vdash a : \{p\}a\{q_1\} \quad \Phi,\Psi \vdash a : \{p\}a\{q_2\}}{\Phi,\Psi \vdash a : \{p\}a\{q_1 \wedge q_2\}}\ (a\text{-}\mathsf{meet}) \qquad \frac{}{\Phi,\Psi \vdash a : \{p\}a\{\mathsf{true}\}}\ (a\text{-}\mathsf{meet}_0)$$

$$\frac{\Phi,\Psi \vdash \phi : \{p_1\}f\{q\} \quad \Phi,\Psi \vdash \phi : \{p_2\}f\{q\}}{\Phi,\Psi \vdash \phi : \{p_1 \vee p_2\}f\{q\}}\ (\mathsf{join}') \qquad \frac{\Phi,\Psi \vdash \phi_1 : \{p \wedge r\}f\{q\} \quad \Phi,\Psi \vdash \phi_2 : \{p \wedge \neg r\}f\{q\}}{\Phi,\Psi \vdash r[\phi_1,\phi_2] : \{p\}f\{q\}}\ (\mathsf{join}'')$$

Table 4. A sound and complete Hoare-style calculus for the synthesis of programs.

take Φ to contain the axioms $\bigvee_{u \in V} p_u$ and $\neg(p_u \wedge p_v)$ for all $u, v \in V$ with $u \neq v$. The axioms of Φ say that the token is on exactly one vertex. So, we can identify the set At_Φ of Φ-consistent atoms with the set $\{p_u \mid u \in V\}$. For every vertex $u \in V$, we introduce an atomic action $u!$, which moves the token to the vertex u. So, we take Ψ to contain the axioms $\{\mathsf{true}\}u!\{p_u\}$ for every $u \in V$. To emphasize that Φ and Ψ depend on G, let us denote them by Φ_G and Ψ_G respectively. For an arbitrary vertex $u \in V$, define the program term (take transition from u) to be equal to $\bigsqcup_{v \in uR} v!$ if $u \in V_0$, and equal to $\bigsqcap_{v \in uR} v!$ if $u \in V_1$. Now, we put

$$\begin{aligned} f_G = \mathsf{while}\,(\textstyle\bigvee\{p_u \mid u \in V \setminus E\})\ \mathsf{do}& \\ \mathsf{if}\ p_u\ \mathsf{then}\ &(\text{take transition from } u) \\ &\cdots \\ \mathsf{else\ if}\ p_w\ \mathsf{then}\ &(\text{take transition from } w) \end{aligned}$$

which describes how the safety game is played. A play stops as soon as an error vertex is encountered.

Theorem 32 (safety games). Let $G = (V_0, V_1, R, E)$ be a finite safety game. Player 0 has a winning strategy from $u \in V_0 \cup V_1$ iff $\Phi_G, \Psi_G \vdash \{p_u\}f_G\{\mathsf{false}\}$.

8 Discussion & Conclusion

At a technical level, the present work is closely related to the line of work on the propositional fragment of Hoare logic, called *Propositional Hoare Logic* or PHL

[6]. In [8,7], a propositional variant of Hoare logic for mutually recursive programs is investigated. The present work differs from both [6,8] in considering the combination of angelic and demonic nondeterminism, which presents significant new challenges for obtaining completeness and decision procedures.

An extension of Propositional Dynamic Logic, called *Game Logic* [12], is also relevant to our work. We note that there are no completeness results for full Game Logic, and that the theory we consider is *not* a fragment of Game Logic. Even though hypotheses-free Hoare assertions $\{p\}f\{q\}$ can be encoded in Dynamic Logic as partial correctness formulas $p \to [f]q$, there is no direct mechanism for encoding the hypotheses of an implication $\Phi, \Psi \Rightarrow \{p\}f\{q\}$ (which would correspond to some kind of global consequence relation in Dynamic Logic).

We have considered here the weak (over the class *All*) and the strong (over the class *Dem*) Hoare theories of dual nondeterminism, and we have obtained sound and unconditionally complete Hoare-style calculi for both of them. We have also shown that they can be both be decided in exponential time, and that the strong Hoare theory is EXPTIME-hard. Finally, we have extended our proof system so that it constructs program terms for the strategies of the angel, thus obtaining a sound and complete calculus for synthesis.

References

1. Back, R.J., Wright, J.: Refinement Calculus: A Systematic Introduction. Springer Heidelberg (1998)
2. Celiku, O., von Wright, J.: Implementing angelic nondeterminism. In: Tenth Asia-Pacific Software Engineering Conference. pp. 176–185 (2003)
3. Cook, S.A.: Soundness and completeness of an axiom system for program verification. SIAM Journal on Computing 7(1), 70–90 (1978)
4. Dijkstra, E.W.: Guarded commands, nondeterminacy and formal derivation of programs. Communications of the ACM 18(8), 453–457 (1975)
5. Hoare, C.A.R.: An axiomatic basis for computer programming. Communications of the ACM 12(10), 576–580,583 (1969)
6. Kozen, D.: On Hoare logic and Kleene algebra with tests. ACM Transactions on Computational Logic 1(1), 60–76 (2000)
7. Mamouras, K.: The Hoare logic of deterministic and nondeterministic monadic recursion schemes (2014), manuscript
8. Mamouras, K.: On the Hoare theory of monadic recursion schemes. In: Proceedings of CSL-LICS (2014)
9. Mamouras, K.: Synthesis of strategies using the Hoare logic of angelic and demonic nondeterminism (2015), in preparation
10. Martin, C.E., Curtis, S.A., Rewitzky, I.: Modelling nondeterminism. In: Mathematics of Program Construction. pp. 228–251 (2004)
11. Morgan, C.: Programming From Specifications. Prentice-Hall (1998)
12. Pauly, M., Parikh, R.: Game logic — An overview. Studia Logica 75(2), 165–182 (2003)
13. Rewitzky, I.: Binary multirelations. In: Theory and Applications of Relational Structures as Knowledge Instruments, pp. 256–271. Springer (2003)
14. Thomas, W.: On the synthesis of strategies in infinite games. In: STACS '95. pp. 1–13 (1995)