Vassar College
Digital Window @ Vassar

Faculty Research and Reports

Winter 2-26-2018

Information Flow under Budget Constraints

Pavel Naumov
pnaumov@vassar.edu

Jia Tao
Lafayette College, taoj@lafayette.edu

Citation Information

Naumov, Pavel and Tao, Jia, "Information Flow under Budget Constraints" (2018). *Faculty Research and Reports*. 111.
https://digitalwindow.vassar.edu/faculty_research_reports/111

Information Flow under Budget Constraints

Pavel Naumov, Vassar College
Jia Tao, Lafayette College

Although first proposed in the database theory as properties of functional dependencies between attributes, Armstrong's axioms capture general principles of information flow by describing properties of dependencies between sets of pieces of information. This article generalizes Armstrong's axioms to a setting in which there is a cost associated with information. The proposed logical system captures general principles of dependencies between pieces of information constrained by a given budget.

CCS Concepts:•**Theory of computation** → **Logic;** *Proof theory;*

General Terms: Theory, Security, Economics

Additional Key Words and Phrases: Armstrong's axioms, axiomatization, completeness, budget constraints

ACM Reference Format:
Pavel Naumov and Jia Tao, 2016. Information Flow under Budget Constraints. *ACM Trans. Comput. Logic* 9, 4, Article 39 (March 2010), 27 pages.
DOI: 0000001.0000001

1. INTRODUCTION

1.1. Functional Dependency

Armstrong [1974] introduced a system of three axioms describing the properties of functional dependencies between sets of attributes in a database. The applicability of these axioms goes far beyond the domain of databases. They capture the properties of functional dependency between any two sets of pieces of information. To describe this setting informally, one can think of an agent that has knowledge of some of the pieces of information and is interested in uncovering some other pieces. For example, knowing a cyphertext c and the decryption key k, one can determine the plain text message m. We write this as $c, k \rhd m$. Yet, one cannot determine the original message from the cyphertext alone without the encryption key, and thus, $\neg(c \rhd m)$. Keeping the intended epistemic interpretation in mind, we refer to the pieces of information as *secrets*.

The property $c, k \rhd m$ is valid when secrets c, k, and m are a cyphertext, a decryption key, and the corresponding plain text message. However, it may not be valid under some other interpretation of these secrets. Armstrong's axioms capture the most general properties of functional dependencies that are valid in all settings. These axioms are:

(A1) *Reflexivity*: $A \rhd B$, if $B \subseteq A$,
(A2) *Augmentation*: $A \rhd B \to A, C \rhd B, C$,
(A3) *Transitivity*: $A \rhd B \to (B \rhd C \to A \rhd C)$,

Author's addresses: P. Naumov, Computer Science Department, Vassar College, Poughkeepsie, NY 12604, email:pnaumov@vassar.edu; Jia Tao, Computer Science Department, Lafayette College, Easton, PA 18042, email:taoj@lafayette.edu.

1529-3785/2010/03-ART39 $15.00
DOI: 0000001.0000001

where A, B denotes the union of sets of secrets A and B, and $\varphi \to \psi$ denotes the logical implication. Armstrong [1974] proved the soundness and the completeness of this logical system with respect to a database semantics.

The above axioms became known in database literature as Armstrong's axioms [Garcia-Molina et al. 2009, p. 81]. Beeri, Fagin, and Howard [1977] suggested a variation of Armstrong's axioms that describes properties of multi-valued dependency. Hartmann, Link, and Schewe [2004] investigated a "weak" version of functional dependency. Väänänen [2007] proposed a first order version of these principles. Naumov and Nicholls [2014] developed a similar set of axioms for what they called the *rationally* functional dependency.

1.2. Approximate Dependency

There have been two different approaches to extending Armstrong's axioms to handle approximate reasoning. Bělohlávek and Vychodil [2006] described a complete logical system that formally captures the relation *approximate values of secrets in set A functionally determine approximate values of secrets in set B*. In [2017], Väänänen considered the relation *secrets in set A determine secrets in set B with exception of p fraction of possible combinations of values of all secrets*. We denote this relation by $A \rhd_p B$. For example, $A \rhd_{0.05} B$ means that secrets in set A determine secrets in set B in all but 5% of the possible combinations. Väänänen [2017] proposed a complete axiomatic system for this relation, consisting of the following principles for all real numbers $p, q \in [0, 1]$:

(1) Reflexivity: $A \rhd_0 B$, where $B \subseteq A$,
(2) Totality: $A \rhd_1 B$,
(3) Weakening: $A \rhd_p C, D \to A, B \rhd_p C$,
(4) Augmentation: $A \rhd_p B \to A, C \rhd_p B, C$,
(5) Transitivity: $A \rhd_p B \to (B \rhd_q C \to A \rhd_{p+q} C)$, where $p + q \leq 1$,
(6) Monotonicity: $A \rhd_p B \to A \rhd_q B$, where $p \leq q$.

Note that Väänänen's relation $A \rhd_p B$, when $p = 0$, is exactly the original Armstrong's functional dependency relation. In the case of an arbitrary p, relation $A \rhd_p B$ could be considered as a "weaker" form of functional dependency, which might hold even in the cases where the functional dependency does not hold.

1.3. Our Contribution

In this article we propose another interpretation of atomic predicate $A \rhd_p B$ that we call *the budget-constrained dependency*. Just like Väänänen's approximate dependency, the budget-constrained dependency is a weaker form of the original Armstrong's functional dependency relation. Intuitively, $A \rhd_p B$ means that *an agent who already knows secrets in set A can recover secrets in set B at cost no more than p*. More formally, we assume that a non-negative cost is assigned to each secret and that $A \rhd_p B$ means that there is a way to add several secrets with the total cost no more than p to set A in such a way that the extended set of secrets functionally determines all secrets in set B.

One example of such a setting is fees associated with information access: criminal background check fees, court records obtaining fees, etc. Another example is geological explorations, where learning about deposits of mineral resources often requires costly drilling. Although it is convenient to think about a budget constraint as a financial one, a budget constraint can also refer to a limit on time, space, or some other resource.

In this article we introduce a sound and complete logical system for the budget-constrained dependency which is based on the following three principles that generalize Armstrong's axioms:

(1) *Reflexivity*: $A \rhd_p B$, if $B \subseteq A$,

(2) *Augmentation*: $A \rhd_p B \to A, C \rhd_p B, C$,
(3) *Transitivity*: $A \rhd_p B \to (B \rhd_q C \to A \rhd_{p+q} C)$.

2. OTHER RELATED LITERATURE

The axiomatic system proposed in this article is related to other logical systems for reasoning about bounded resources. The classical logical system for reasoning about resources is the linear logic of Girard [1987]. Alechina and Logan [2002] presented a family of logical systems for reasoning about beliefs of a perfect reasoner that can only derive consequences of her beliefs after some time delay. This approach has been further developed into the multi-agent Timed Reasoning Logic in [Alechina et al. 2004]. Bulling and Farwer [2010] proposed Resource-Bounded Tree Logics for reasoning about resource-bounded computations and obtained results on the complexity and decidability of model checking for these logics. Alechina, Logan, Nguyen, and Rakib [2011] incorporated resource requirements into Coalition logic and gave a sound and complete axiomatization of the resulting system. Another logical system for reasoning about knowledge under bounded resources was proposed by Jamroga and Tabatabaei [2013]. Their paper focuses on the expressive power of the language of the system and the model checking algorithm. Naumov and Tao introduced sound and complete modal logics for reasoning about budget-constrained knowledge [2015] and cost of privacy [2016b]. Unlike our current system all of the above logics do not provide a language for expressing functional dependencies. An extended abstract of the present work, without the proof of the completeness, has previously appeared as [Naumov and Tao 2016a].

3. OUTLINE

The rest of the article is organized as follows. In Section 4 we informally discuss several properties of budget-constrained dependencies that illustrate the challenges of adopting Armstrong axioms to this new setting. In Section 5.1 we formally define the language of our logical system and its informational semantics. In Section 5.2 we list the axioms of the system that have already been discussed in the introduction. In Section 5.3 we give several examples of formal proofs in our logical system. In Section 6 we prove the soundness of our axioms with respect to the informational semantics. Section 7 states the completeness theorem. In Section 8 we introduce an auxiliary hypergraph semantics of our logical system and prove the completeness with respect to this semantics. In Section 9 we use the result obtained in the previous section to prove the completeness of our logical system with respect to the informational semantics. Section 10 strengthens the informational completeness results by proving the completeness with respect to a more narrow class of *finite cost* informational models. Section 11 concludes the article.

4. MOTIVATIONAL EXAMPLES

Armstrong's axioms of functional dependency as well as our axioms of budget-constrained functional dependency can be formulated into two different ways using languages with different expressive power.

One approach is to allow only statements in our language that have the form $A \rhd_p B$ and not allow Boolean combinations of such statements. In this case, Armstrong's axioms should be stated as inference rules that allow to derive statements of the form $A \rhd_p B$ from other statements of the same form.

The other approach is to include Boolean connectives into the language. In this case, statements of the form $A \rhd_p B$ become atomic statements in the language. Then, Armstrong's axioms can be stated as actual axioms that would be used in the logical system along with the propositional tautologies and Modus Ponens inference rule.

The second approach clearly yields a more expressive language. While the proofs of the completeness of original Armstrong's axioms of functional dependency are surprisingly similar for these two cases, the situation is different when it comes to budget-constrained dependency. The more expressive language requires a significantly more sophisticated argument to prove the completeness theorem. In this article we only consider the more expressive language. In the rest of this section we look at several examples to compare challenges raised by the proofs of the completeness for Armstrong's functional dependency and our budget-constrained dependency.

As the first example, consider the formula $a \rhd b \to b \rhd a$ in the language without budget constraints. To construct a counterexample for this formula we need to describe a model in which secret a functionally determines secret b but not vice versa. Informally, to construct this model, imagine a and b to be two paper folders. Let folder a contain copies of two different (and unrelated to each other) documents: X and Y, and let folder b contain only a copy of document Y. In this case, an agent can recover the content of folder b based on folder a but not vice versa.

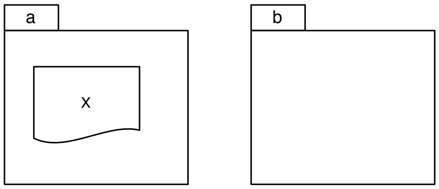

Fig. 1. Formula $a \rhd b$ is true, but formula $b \rhd a$ is false.

There is even a simpler counterexample for formula $a \rhd b \to b \rhd a$. Namely, consider a model in which folder a stores a copy of document X and folder b is empty, see Figure 1. In this model, based on the content of folder a one can vacuously recover the content of empty folder b. At the same time, based on the content of empty folder b one cannot recover the content of folder a. Thus, in this model formula $a \rhd b \to b \rhd a$ is false.

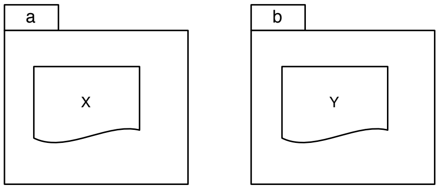

Fig. 2. Formulas $a \rhd b$ and $b \rhd a$ are both false.

Now consider formula $a \rhd b \vee b \rhd a$. To construct its counterexample, one can consider a model in which folders a and b containing copies of two different (and unrelated to each other) documents X and Y respectively, see Figure 2.

To construct counterexamples for more complicated formulas, one can consider models with multiple folders containing copies of multiple documents. An example of such a model is depicted in Figure 3. In this model $a, b \rhd c$ is true because anyone with access to folders a and b knows the content of folder c. The folder/document model informally described here is sufficiently general to create a counterexample for each formula unprovable from Armstrong's axioms.

In fact, the original Armstrong's proof of the completeness for his rule-based system and the proof of the completeness for the corresponding axiom-based system [Heckle

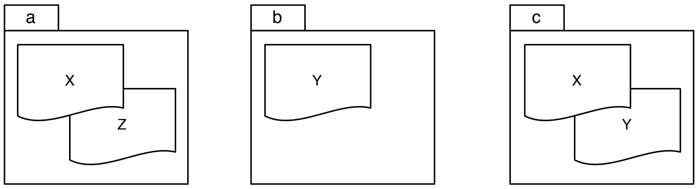

Fig. 3. Formula $a, b \rhd c$ is true.

and Naumov 2014] could be viewed as formalizations of this folder/document construction.

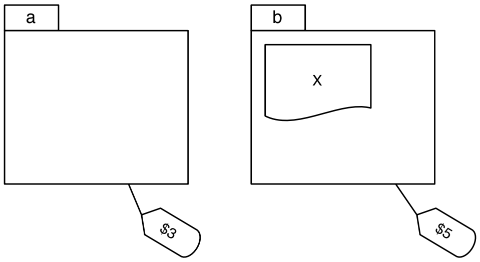

Fig. 4. Formula $a \rhd_4 b$ is false.

The situation becomes significantly more complicated once the cost of information is added to the language. Let us start with a very simple example. If we want to construct a counterexample for formula $a \rhd_4 b$, then we can consider a model depicted in Figure 4 with two folders: a and b, priced at $\$3$ and $\$5$, respectively. The first folder is empty and the second contains a copy of the document X. It is clear that in this model anyone who knows the content of folder a still needs to spend $\$5$ to learn the content of folder b. Thus, budget-constrained dependency $a \rhd_p b$ is not satisfied in this model for each $p < 5$.

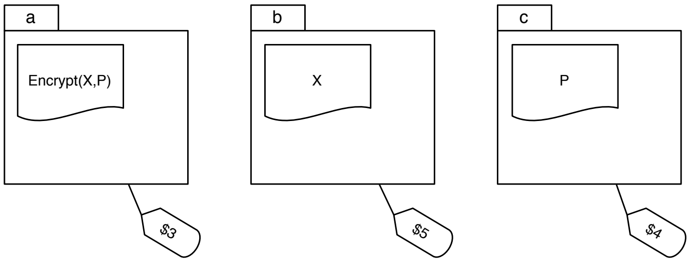

Fig. 5. Formula $a \rhd_4 b \to \varnothing \rhd_4 b$ is false.

Let us now consider a more interesting example. Suppose that we want to construct a counterexample for the formula $a \rhd_4 b \to \varnothing \rhd_4 b$. That is, we want to construct a model where anyone who knows the content of folder a can reconstruct the content of folder b after spending at most $\$4$. Yet, the same cannot be done without access to folder a. To construct such a model we use the cryptographic tool called one-time encryption pad[1]. Our model consists of three folders a, b, and c priced at \$3, \$5, and \$4, respectively, see Figure 5. Let folder b contain a copy of a document X, folder c contain an encryption pad P, and folder a contain the encrypted version of the document. In this model, $\varnothing \rhd_4 b$ is false because \$4 buys either access to the encryption pad in folder c or access to the encrypted text in folder a, but not both. However, formula $a \rhd_4 b$ is true in the same model because anyone who knows encrypted text $Encrypt(X, P)$ can spend \$4 on pad P, decode message X, and thus, learn the content of folder b.

The one-time pad encryption is known in cryptography as a symmetric-key algorithm because the same key (i.e. the one-time pad) could be used to encrypt and to decrypt the text. As a result, in the model depicted in Figure 5, not only formula $a \rhd_4 b$ is true, but formula $b \rhd_4 a$ is true as well.

For the next example, we construct a counterexample for formula

$$a \rhd_4 b \to (\varnothing \rhd_4 b \vee b \rhd_4 a).$$

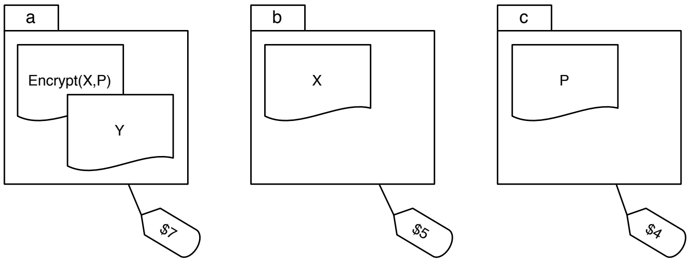

Fig. 6. Formula $a \rhd_4 b \to (\varnothing \rhd_4 b \vee b \rhd_4 a)$ is false.

This is an easier task than one might think because one just needs to modify the previous model by adding to the folder a some extra document not related to the document X and to raise the price of this folder, see Figure 6. This guarantees that the only way to learn all the content of folder a is to buy folder a directly.

The situation becomes much more complicated if we want (i) the value of secret a to be recoverable from the value of secret b and (ii) the value of secret b to be recoverable from the value of secret a, but at a different price. For instance, if we want to construct a counterexample for the following formula:

$$a \rhd_1 b \wedge b \rhd_5 a \to (\varnothing \rhd_5 a \vee \varnothing \rhd_1 b \vee b \rhd_4 a). \tag{1}$$

At first glance, this goal could be achieved using asymmetric key cryptography, commonly used in the public-key encryption. For instance, suppose that folder a contains a document X and folder b contains the same document encrypted with an encryption key k_e, see Figure 7. To obtain the content of folder b based on the content of folder a, one only needs to know the encryption key k_e. To restore the content of folder a based on folder b one needs to know the value of the decryption[2] key k_d. If the encryption key and the decryption key are priced at \$1 and \$5 respectively, the formula $b \rhd_4 a$ is not

[1] The one-time encryption pad is not the only way to construct a counterexample for formula $a \rhd_4 b \to \varnothing \rhd_4 b$. We introduce one-time pads to prepare readers for the general proof of the completeness presented later in this article.

[2] In public-key cryptography, an encryption key is known as the public key and a decryption key as the private key. We do not use these terms here because in our setting neither of the keys is public in the sense that both of them have associated costs.

satisfied from the cryptographic point of view. Since folders a and b are priced in this model at $100 each, formulas $\varnothing \rhd_5 a$ and $\varnothing \rhd_1 b$ are not satisfied either. Thus, the entire formula (1) is not satisfied from the cryptographic point of view.

Note, however, that cryptographic asymmetric-key algorithms are only polynomial time secure and the proof of polynomial time security requires an appropriate computational hardness assumption [Katz 2010, Ch. 2]. In other words, in public-key cryptography, the encrypted text can be decrypted using only the public encryption key if one has exponential time for the decryption. Neither Armstrong's [Armstrong 1974] definition of functional dependency nor our definition of budget-constrained functional dependency, given in Definition 5.6 below, assumes any upper bound on the computability of the functional dependency. From our point of view, one would be able to eventually restore the content of folder a based on folder b by spending $1 on the content of folder c. Thus, in the above setting, without the polynomial restriction on computability, not only formula $b \rhd_4 a$ is true, but formula $b \rhd_1 a$ is true as well.

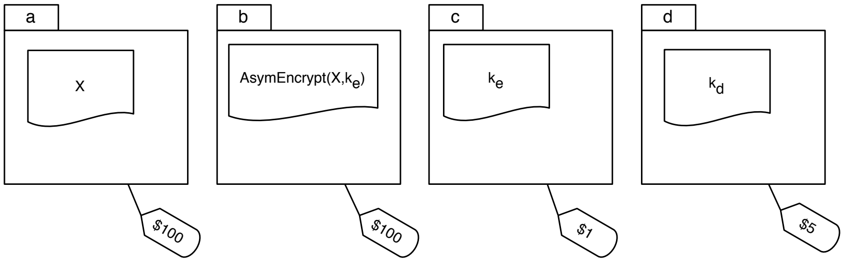

Fig. 7. An asymmetric key model.

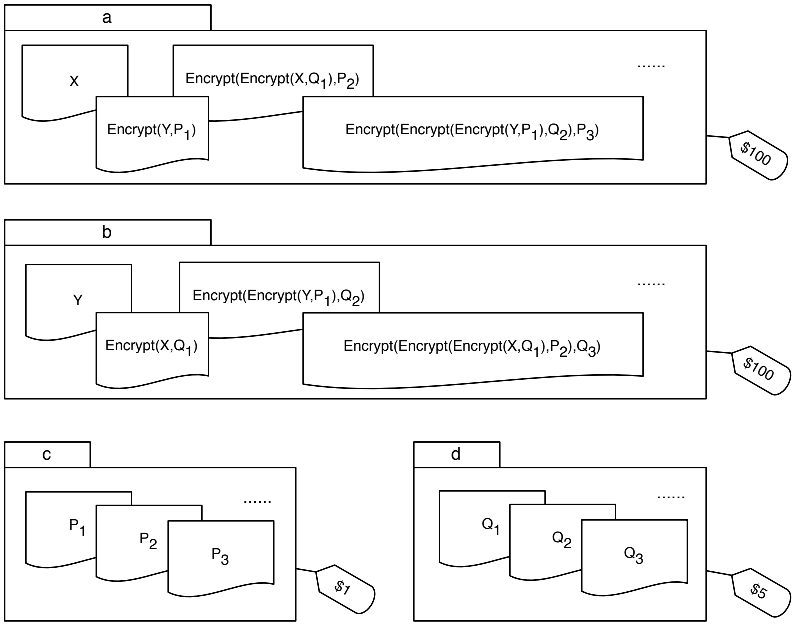
a
X
Encrypt(Encrypt(X,Q_1),P_2)
......
Encrypt(Y,P_1)
Encrypt(Encrypt(Encrypt(Y,P_1),Q_2),P_3)
$100
b
Y
Encrypt(Encrypt(Y,P_1),Q_2)
......
Encrypt(X,Q_1)
Encrypt(Encrypt(Encrypt(X,Q_1),P_2),Q_3)
$100
c
P_1
P_2
P_3
......
$1
d
Q_1
Q_2
Q_3
......
$5

Fig. 8. A counterexample for statement (1).

Figure 8 shows a counterexample for statement (1) that uses non-computable functional dependency. Assume that folders a and b contain copies of unrelated documents X and Y, folder c contains an infinite supply of one-time encryption pads $P_1, P_2, P_3, \dots$ and folder d contains another infinite set of one-time encryption pads $Q_1, Q_2, Q_3, \dots$. First, encrypt document Y with one-time pad P_1 and place a copy of the resulting cyphertext $Encrypt(Y, P_1)$ into folder a. Next, encrypt $Encrypt(Y, P_1)$ with pad Q_2 and place a copy of the resulting cyphertext $Encrypt(Encrypt(Y, P_1), Q_2)$ into folder b. Then, use pad P_3 to encrypt $Encrypt(Encrypt(Y, P_1), Q_2)$ and place a copy of the resulting cyphertext $Encrypt(Encrypt(Encrypt(Y, P_1), Q_2), P_3)$ into folder a, and so on ad infinitum. Perform similar steps with the document X, as shown in Figure 8.

To show that the model depicted in Figure 8 is a counterexample for formula (1), we need to prove that both formulas $a \rhd_1 b$ and $b \rhd_5 a$ are satisfied in this model and each of the formulas $\varnothing \rhd_5 a$, $\varnothing \rhd_1 b$, and $b \rhd_4 a$ is not satisfied. First, notice that formula $a \rhd_1 b$ is satisfied because folder a contains all documents in folder b encrypted with one-time pads $P_1, P_2, \dots$ and that all these pads could be acquired for $1 by buying folder c. Second, formula $b \rhd_5 a$ is satisfied for a similar reason using pads $Q_1, Q_2, \dots$. Third, formula $\varnothing \rhd_5 a$ is not satisfied because for $5 one can only buy either folder c or folder d, both containing only one-time pads. In the absence of folder b, one-time encryption pads can not be used to recover document X stored in folder a. Formula $\varnothing \rhd_1 b$ is not satisfied for a similar reason. Finally, $b \rhd_4 a$ is not satisfied because $4 is not enough to buy the content of folder d. This amount of money can only be used to buy pads $P_1, P_2, \dots$ in folder c. Knowing the content of folder b and one-time pads $P_1, P_2, \dots$, one can not recover document X contained in folder a. The counterexample described above produces non-computable functional dependency because the number of folders is infinite.

In this article we prove the completeness of our logical system. At the core of this proof is a generalized version of the construction presented in Figure 8.

5. LOGICAL SYSTEM

5.1. Syntax and Semantics

In this section we introduce the language of our system and formally describe its intended semantics that we call *informational semantics*. Later on we introduce an auxiliary *hypergraph semantics* used as a technical tool in the proof of the completeness with respect to the original informational semantics.

Definition 5.1. For any set of "secrets" $\mathcal{S}$, let language $\Phi(\mathcal{S})$ be the minimum set of formulas such that

(1) $A \rhd_p B \in \Phi(\mathcal{S})$ for all finite sets $A, B \subseteq \mathcal{S}$ and all real numbers $p \geq 0$,
(2) if $\varphi \in \Phi(\mathcal{S})$, then $\neg\varphi \in \Phi(\mathcal{S})$,
(3) if $\varphi, \psi \in \Phi(\mathcal{S})$, then $\varphi \rightarrow \psi \in \Phi(\mathcal{S})$.

Next, we introduce the formal informational semantics of our logical system. The only significant difference between our semantics and the one used by Armstrong [1974] is the cost function $\|\cdot\|$ that assigns a non-negative cost to each secret. Note that we assume that the cost is assigned to a secret, not to its value. For example, if we assign a certain cost to a folder with documents, then this cost is uniform and does not depend on the content of the documents in this folder.

Definition 5.2. An informational model is a tuple $\langle \mathcal{S}, \{D_a\}_{a\in\mathcal{S}}, \|\cdot\|, \mathcal{L}\rangle$, where

(1) $\mathcal{S}$ is an arbitrary set of "secrets",
(2) D_a is a set representing the domain of secret $a \in \mathcal{S}$,

(3) $\|\cdot\|$ is a cost function that maps each secret $a \in \mathcal{S}$ into a non-negative real number or infinity $+\infty$,
(4) $\mathcal{L} \subseteq \prod_{a \in \mathcal{S}} D_a$ is the set of vectors of values of secrets that satisfy the constraints imposed by the informational model.

Note that we allow infinite attribute costs in our semantics captured in Definition 5.2 to include the possibility of attributes that cannot be bought. For the same reason, we do *not* allow infinite costs in our syntax given in Definition 5.1. In the example depicted in Figure 8, folders are secrets and the information stored in the documents contained in a folder is a value of such a secret. The set of all possible values of a secret is its domain. The cost of different secrets is specified explicitly in Figure 8. Note that there is a certain dependency between the plaintext, one-time encryption pads, and the cyphertext. In other words, not all combinations of values of different secrets are possible. The set $\mathcal{L}$ is the set of all possible combinations of these values.

We allow the cost $\|a\|$ of a secret a to be infinity. Informally, one can interpret this as secret a not being available for purchase at any cost. If all secrets are available for sale, then we say that the informational model is *finite cost*.

As an example, the setting described in Figure 4 could be formally captured in informational model $I_0 = \langle \{a, b\}, \{D_x\}_{x \in \{a,b\}}, \|\cdot\|, \mathcal{L}\rangle$, where the domain (range of values) D_a of the empty folder a is a single element set: $\{null\}$, the domain D_b of secret b is the set of all, say binary, strings $\{0,1\}^*$, cost $\|a\|$ of secret a is 3, cost $\|b\|$ of secret b is 5, and set of vectors $\mathcal{L}$ is the set of all pairs in set $D_a \times D_b = \{\langle null, s\rangle \mid s \in \{0,1\}^*\}$.

Definition 5.3. Informational model $\langle \mathcal{S}, \{D_a\}_{a \in \mathcal{S}}, \|\cdot\|, \mathcal{L}\rangle$ is finite cost if $\|a\| < +\infty$ for each $a \in \mathcal{S}$.

For example, informational model I_0 for the setting described in Figure 4 is a finite cost model, because $\|a\| = 3 < \infty$ and $\|b\| = 5 < \infty$.

Definition 5.4. For any vector $\ell_1 = \langle f_a^1 \rangle_{a \in \mathcal{S}} \in \mathcal{L}$, any vector $\ell_2 = \langle f_a^2 \rangle_{a \in \mathcal{S}} \in \mathcal{L}$, and any set $A \subseteq \mathcal{S}$, let $\ell_1 =_A \ell_2$ if $f_a^1 = f_a^2$ for each secret $a \in A$.

For example, $\langle null, 01\rangle =_{\{a\}} \langle null, 1011\rangle$ and $\langle null, 01\rangle \neq_{\{a,b\}} \langle null, 1011\rangle$ for model I_0.

Definition 5.5. For each finite set $A \subseteq \mathcal{S}$, let $\|A\| = \sum_{a \in A} \|a\|$.

Thus, $\|\{a, b\}\| = \|a\| + \|b\| = 3 + 5 = 8$ for model I_0.

The next definition is the key definition of this section. It specifies the formal semantics of our logical system. Item 1. of this definition provides the exact meaning of the budget-constrained dependency. In this definition and throughout the rest of the article, by A, B we denote the union of sets A and B.

Definition 5.6. For each informational model $I = \langle \mathcal{S}, \{D_a\}_{a \in \mathcal{S}}, \|\cdot\|, \mathcal{L}\rangle$ and each formula $\varphi \in \Phi(\mathcal{S})$, the satisfiability relation $I \vDash \varphi$ is defined as follows:

(1) $I \vDash A \rhd_p B$ when there is a finite set $C \subseteq \mathcal{S}$ such that $\|C\| \leq p$ and for each pair of vectors $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{A,C} \ell_2$, then $\ell_1 =_B \ell_2$,
(2) $I \vDash \neg\psi$ if $I \nvDash \psi$,
(3) $I \vDash \psi \to \chi$ if $I \nvDash \psi$ or $I \vDash \chi$.

Then, $I_0 \Vdash \varnothing \rhd_0 a$ because $\ell_1 =_a \ell_2$ for any two vectors $\ell_1, \ell_2 \in \{null\} \times \{0,1\}^*$.

5.2. Axioms

For any set of secrets $\mathcal{S}$, our logical system, in addition to the propositional tautologies in language $\Phi(\mathcal{S})$ and the Modus Ponens inference rule, contains the following axioms:

(1) Reflexivity: $A \rhd_p B$, where $B \subseteq A$,

(2) Augmentation: $A \rhd_p B \to A, C \rhd_p B, C$,
(3) Transitivity: $A \rhd_p B \to (B \rhd_q C \to A \rhd_{p+q} C)$.

We write $\vdash \varphi$ if formula φ is derivable in our system. Also, we write $X \vdash \varphi$ if formula φ is derivable in our system extended by the set of additional axioms X.

5.3. Examples of Proofs

We prove the soundness of our logical system in the next section. Here we provide several examples of formal proofs in this system. We start by showing that Weakening and Monotonicity axioms [Väänänen 2017] are derivable in our system.

PROPOSITION 5.7 (WEAKENING). $\vdash A \rhd_p C, D \to A, B \rhd_p C$.

PROOF. By Augmentation axiom,

$$\vdash A \rhd_p C, D \to A, B \rhd_p B, C, D. \tag{2}$$

By Reflexivity axiom,

$$\vdash B, C, D \rhd_0 C. \tag{3}$$

By Transitivity axiom,

$$A, B \rhd_p B, C, D \to (B, C, D \rhd_0 C \to A, B \rhd_p C). \tag{4}$$

Finally, from (2), (3), and (4), by the laws of propositional logic,

$$\vdash A \rhd_p C, D \to A, B \rhd_p C.$$

□

PROPOSITION 5.8 (MONOTONICITY). $\vdash A \rhd_p B \to A \rhd_q B$, *where* $p \le q$.

PROOF. By Reflexivity axiom,

$$\vdash B \rhd_{q-p} B. \tag{5}$$

By Transitivity axiom,

$$\vdash A \rhd_p B \to (B \rhd_{q-p} B \to A \rhd_q B). \tag{6}$$

Finally, from (5) and (6), by the laws of propositional logic,

$$\vdash A \rhd_p B \to A \rhd_q B.$$

□

As our last example, we prove a generalized version of Augmentation axiom.

PROPOSITION 5.9. $\vdash A \rhd_p B \to (C \rhd_q D \to A, C \rhd_{p+q} B, D)$.

PROOF. By Augmentation axiom,

$$\vdash A \rhd_p B \to A, C \rhd_p B, C \tag{7}$$

and

$$\vdash C \rhd_q D \to B, C \rhd_q B, D. \tag{8}$$

At the same time, by Transitivity axiom,

$$\vdash A, C \rhd_p B, C \to (B, C \rhd_q B, D \to A, C \rhd_{p+q} B, D). \tag{9}$$

Finally, from (7), (8), and (9), by the laws of propositional logic,

$$\vdash A \rhd_p B \to (C \rhd_q D \to A, C \rhd_{p+q} B, D).$$

□

6. SOUNDNESS

In this section we prove the soundness of our logical system.

THEOREM 6.1. *If $\varphi \in \Phi(\mathcal{S})$ and $\vdash \varphi$, then $I \vDash \varphi$ for each informational model $I = \langle \mathcal{S}, \{D_a\}_{a \in \mathcal{S}}, \|\cdot\|, \mathcal{L}\rangle$.*

The soundness of propositional tautologies and the Modus Ponens inference rule follows from Definition 5.6 in the standard way. Below we prove the soundness of the remaining axioms as separate lemmas.

LEMMA 6.2. *For all finite sets $A, B \subseteq \mathcal{S}$, if $B \subseteq A$, then $I \vDash A \rhd_p B$.*

PROOF. Let $C = \varnothing$. Thus, $\|C\| = \|\varnothing\| = 0 \le p$. Consider any two vectors $\ell_1, \ell_2 \in \mathcal{L}$ such that $\ell_1 =_{A,C} \ell_2$. It suffices to show that $\ell_1 =_B \ell_2$, which is true due to Definition 5.4 and the assumption $B \subseteq A$. □

LEMMA 6.3. *For all finite sets $A, B, C \subseteq \mathcal{S}$, if $I \vDash A \rhd_p B$, then $I \vDash A, C \rhd_p B, C$.*

PROOF. By Definition 5.6, assumption $I \vDash A \rhd_p B$ implies that there is a set $D \subseteq \mathcal{S}$ such that (i) $\|D\| \le p$ and (ii) for each $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{A,D} \ell_2$, then $\ell_1 =_B \ell_2$.

Consider now $\ell_1, \ell_2 \in \mathcal{L}$ such that $\ell_1 =_{A,C,D} \ell_2$. It suffices to show that $\ell_1 =_{B,C} \ell_2$. Note that assumption $\ell_1 =_{A,C,D} \ell_2$ implies that $\ell_1 =_{A,D} \ell_2$ and $\ell_1 =_C \ell_2$ by Definition 5.4. Due to condition (ii) above, the former implies that $\ell_1 =_B \ell_2$. Finally, statements $\ell_1 =_B \ell_2$ and $\ell_1 =_C \ell_2$ together imply that $\ell_1 =_{B,C} \ell_2$. □

LEMMA 6.4. *For all finite sets $A, B, C \subseteq \mathcal{S}$, if $I \vDash A \rhd_p B$ and $I \vDash B \rhd_q C$, then $I \vDash A \rhd_{p+q} C$.*

PROOF. By Definition 5.6, assumption $I \vDash A \rhd_p B$ implies that there is $D_1 \subseteq \mathcal{S}$ such that (i) $\|D_1\| \le p$ and (ii) for each $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{A,D_1} \ell_2$, then $\ell_1 =_B \ell_2$.

Similarly, assumption $I \vDash B \rhd_q C$ implies that there is $D_2 \subseteq \mathcal{S}$ such that (iii) $\|D_2\| \le q$ and (iv) for each $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{B,D_2} \ell_2$, then $\ell_1 =_C \ell_2$.

Let $D = D_1, D_2$. By Definition 5.5, $\|D\| \le \|D_1\| + \|D_2\|$. Taking into account statements (i) and (iii) above, we conclude that $\|D\| \le p + q$. Consider any two vectors $\ell_1, \ell_2 \in \mathcal{L}$ such that $\ell_1 =_{A,D} \ell_2$. It suffices to show that $\ell_1 =_C \ell_2$. Indeed, by Definition 5.4, assumption $\ell_1 =_{A,D} \ell_2$ implies that $\ell_1 =_{A,D_1} \ell_2$. Hence, $\ell_1 =_B \ell_2$ due to condition (ii). At the same time, assumption $\ell_1 =_{A,D} \ell_2$ also implies that $\ell_1 =_{D_2} \ell_2$ by Definition 5.4. Thus, $\ell_1 =_{B,D_2} \ell_2$ by Definition 5.4. Therefore, $\ell_1 =_C \ell_2$ due to condition (iv). □

This concludes the proof of Theorem 6.1.

7. ON THE COMPLETENESS THEOREM

The main result of this article is a completeness theorem for our logical system with respect to the informational semantics. The completeness could be stated in different non-equivalent forms that we discuss and compare in this section.

Informally, a completeness theorem states that if a formula φ is not provable in our system, then there is an informational model I such that $I \nvDash \varphi$. To state the theorem formally, we need to decide if model I must use only secrets explicitly mentioned in formula φ or a set of secrets of model I could be a superset of the set of secrets used in formula φ.

This distinction applies not only to our system, but to other logical systems as well. For example, formulas in first order logic can have constants. When we prove the completeness of the first order logic, we allow universes that have more elements than the number of constants. This is significant because, for example, formula

$$\forall x(c_1 = c_2 \vee x = c_1 \vee x = c_2) \tag{10}$$

is not provable in the first order logic, but it is true in any model with a universe consisting of only elements that are interpretations of c_1 and c_2. Thus, to construct a counterexample for this formula one needs to consider first order models with more than two elements in the universe.

The situation with our logical system is similar. To prove the completeness of the system we often need to introduce additional secrets not explicitly mentioned in the formula. An analog of formula (10) is, for example, formula

$$\neg(a \rhd_1 b) \to (\neg(b \rhd_1 a) \to \neg(\varnothing \rhd_2 a, b)). \tag{11}$$

This formula is true in any informational model that has only two secrets explicitly mentioned in the formula: secret a and secret b. Indeed, the assumption $\neg(a \rhd_1 b)$ implies that costs of secret b is more than 1. Similarly, assumption $\neg(b \rhd_1 a)$ implies that costs of secret a is more than 1. So, given a budget of only 2, one can buy at most one of secrets a and b. Without loss of generality, assume that secret a is bought. After that purchase, the amount left is less than 1. Per assumption $\neg(a \rhd_1 b)$, the value of b is not attainable on this budget.

At the same time, formula (11) is not true in the information model that has three secrets: a, b, and c, all priced at 1.5, where values of a and b are unrelated and c is pair $\langle a, b\rangle$. One can think about this example as a formalization of "buy one, get one free" marketing.

In this article we study the most general logical principles of budget-constrained dependency. Thus, we do not include principles like formula (11), that are true only for a specific set of secrets. In other words, when constructing a counterexample for the completeness theorem, we allow additional secrets that are not explicitly mentioned in the original formula. The completeness theorem is stated below.

THEOREM 7.1. *For each formula $\varphi \in \Phi(\mathcal{S})$, if $\nvdash \varphi$, then there is an informational model $I = \langle \mathcal{S}, \{D_a\}_{a\in\mathcal{S}}, \|\cdot\|, \mathcal{L}\rangle$ such that $I \nvDash \varphi$.*

We prove this theorem in Section 9. Later, see Theorem 10.6, we also state and prove the completeness theorem for finite cost informational models.

8. AUXILIARY HYPERGRAPH SEMANTICS

The main goal of the rest of the article is to prove the completeness of our logical system with respect to the informational semantics. To achieve this goal we introduce the *hypergraph* semantics of our logical system and prove that the following statements are equivalent for each formula φ:

(1) φ is provable in our logical system,
(2) φ is satisfied in each informational model,
(3) φ is satisfied in each hypergraph.

We prove the equivalence of these statements by showing that the first statement implies the second, the second implies the third, and the third implies the first. Note that we have already proved in Theorem 6.1 that the first statement implies the second one. In the rest of the article we prove that the second statement implies the third one and that the third one implies the first one. Together, these results will imply the soundness and the completeness of our logical system with respect to the informational semantics.

8.1. Hypergraph Terminology

Before defining the hypergraph semantics of our logical system, we introduce the basic hypergraph terminology used throughout the rest of the article. In mathematics, a hypergraph is a generalization of a graph in which edges have arbitrary numbers of ends, see [Berge 1989]. Our hypergraph semantics is based on weighted directed hypergraphs. In such hypergraphs, edges are directed in the sense that they have multiple tails and multiple heads. For any given edge e, we denote these sets by $in(e)$ and $out(e)$. The edges are weighted in the sense that there is a non-negative value assigned to each edge.

Definition 8.1. A weighted directed hypergraph, or just a "hypergraph", is a tuple $\langle V, E, in, out, w\rangle$, where

(1) V is an arbitrary finite set of "vertices",
(2) E is an arbitrary (possibly infinite) set of "edges", disjoint with set V,
(3) in is a function that maps each edge $e \in E$ into set $in(e) \subseteq V$,
(4) out is a function that maps each edge $e \in E$ into set $out(e) \subseteq V$,
(5) w is a function that maps each edge $e \in E$ into a real number $w(e) \geq 0$.

Definition 8.2. For any hypergraph $\langle V, E, in, out, w\rangle$ and any set $F \subseteq E$, let $w(F) = \sum_{e\in F} w(e)$.

Next, we define the closure A^*_F of a set of vertices A with respect to a set of edges F of a hypergraph. Informally, the closure A^*_F is the set of all vertices that are reachable from a vertex in set A following the directed edges in set F. In order to follow a directed edge $e \in F$, one needs to be able to first reach all vertices in set $in(e)$. To define the closure A^*_F, we first define the partial closure A^k_F of all vertices reachable from A through F in no more than k steps:

Definition 8.3. For each weighted directed hypergraph $\langle V, E, in, out, w\rangle$, each set $A \subseteq V$, each set $F \subseteq E$, and each non-negative integer k, let set A^k_F be defined recursively as follows:

(1) $A^0_F = A$,
(2) for any $k \geq 0$,

$$A^{k+1}_F = A^k_F \cup \bigcup_{\{f\in F \mid in(f)\subseteq A^k_F\}} out(f).$$

Figure 9 depicts a hypergraph where vertices are represented by circles and edges by ovals. We use arrows to indicate tails and heads of an edge. An arrow from a vertex to an edge indicates that the vertex is a tail of the edge, and an arrow from an edge to a vertex indicates that the vertex is a head of the edge. For example, $in(e_1) = \{v_1, v_2\}$ and $out(e_1) = \{v_3, v_4\}$. The same figure shows partial closures $A^0_F = \{v_1, v_2\}$, $A^1_F = \{v_1, v_2, v_3, v_4\}$, and $A^2_F = \{v_1, v_2, v_3, v_4, v_5, v_6\}$, where $A = \{v_1, v_2\}$ and $F = \{e_1, e_2\}$.

Finally, we define closure A^*_F to be the union of all partial closures:

Definition 8.4. $A^*_F = \bigcup_{k\geq 0} A^k_F$.

Next, we establish two properties of closures that are used later in the proof of the completeness for the hypergraph semantics.

LEMMA 8.5. *For each set of vertices $A \subseteq V$ and each set of edges $F \subseteq E$ there is $k \geq 0$ such that $A^*_F = A^k_F$.*

PROOF. By Definition 8.1, the set of all vertices V is finite. Thus, by Definition 8.3, chain $A^0_F \subseteq A^1_F \subseteq A^2_F \subseteq \dots$ is a non-decreasing chain of subsets of finite set V. Hence,

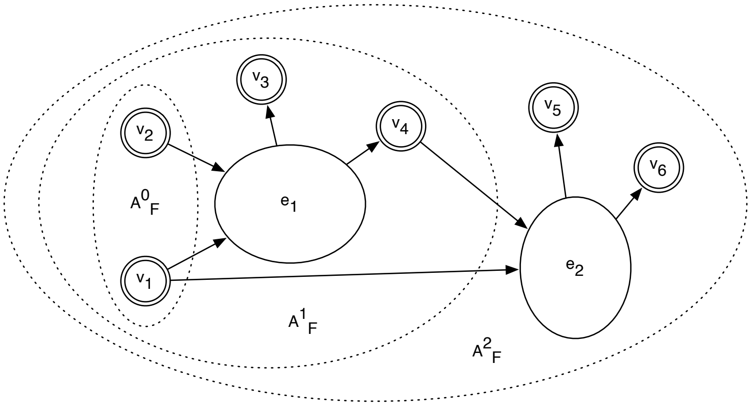

Fig. 9. A_F^k for $A = \{v_1, v_2\}$ and $F = \{e_1, e_2\}$.

there must exist $k \geq 0$ such that all sets in this chain starting with set A_F^k are equal. Therefore, $A_F^* = A_F^k$ by Definition 8.4. □

LEMMA 8.6. *For each $k \geq 0$, each set of vertices $A \subseteq V$, and each set of edges $F \subseteq E$, there is a sequence $A = A_1, f_1, A_2, f_2, \ldots, A_{n-1}, f_{n-1}, A_n = A_F^k$ such that*

(*1*) $n \geq 1$,
(*2*) $f_1, \ldots, f_{n-1} \in F$ *are distinct edges,*
(*3*) $A_1, \ldots, A_{n-1}$, *and* A_n *are subsets of set* V,
(*4*) $in(f_i) \subseteq A_i$, *for each* $1 \leq i < n$,
(*5*) $A_i \cup out(f_i) = A_{i+1}$, *for each* $1 \leq i < n$.

PROOF. We prove this lemma by induction on k. If $k = 0$, then $A_F^k = A$ by Definition 8.3. Therefore, the single-element sequence A is the desired sequence.

For the induction step, assume that there is a sequence

$$A = A_1, f_1, A_2, f_2, \ldots, A_{n-1}, f_{n-1}, A_n = A_F^k$$

that satisfies the conditions 1 through 5 above. Let $g_1, \ldots, g_m$ be all such edges $g \in F$ that $in(g) \subseteq A_F^k$ and $out(g) \nsubseteq A_F^k$. By the condition 5 above, the condition $out(g) \nsubseteq A_F^k$ implies that none of $g_1, \ldots, g_m$ is equal to any of $f_1, \ldots, f_{n-1}$. Note that $A_F^{k+1} = A_F^k \cup \bigcup_{i=1}^m out(g_i)$ by Definition 8.3. Therefore, the two-line sequence

$$A = A_1, f_1, A_2, f_2, \ldots, A_{n-1}, f_{n-1}, A_n = A_F^k, g_1, A_F^k \cup out(g_1),$$
$$g_2, A_F^k \cup out(g_1) \cup out(g_2), g_3, \ldots, g_m, A_F^k \cup \bigcup_{i=1}^{m} out(g_i) = A_F^{k+1}$$

is the required sequence for $k + 1$. □

8.2. Hypergraph Completeness

In this section we define the hypergraph semantics of our logical system and prove the completeness of the system with respect to this auxiliary semantics. In other words, using statements defined in the beginning of Section 8, we prove that the third statement implies the first one. The hypergraph semantics is specified in the following definition. Item 1 in this definition is the key part because it specifies the meaning of the atomic predicate $A \rhd_p B$. Note that we use symbol $\vDash$ for the satisfiability relation under the informational semantics discussed previously and $\Vdash$ for the satisfiability relation under the hypergraph semantics introduced here.

Definition 8.7. For each hypergraph $H = \langle V, E, in, out, w\rangle$ and each formula $\varphi \in \Phi(V)$, the satisfiability relation $H \Vdash \varphi$ is defined as follows:

(1) $H \Vdash A \rhd_p B$ if there is a finite set $F \subseteq E$ such that $w(F) \leq p$ and $B \subseteq A^*_F$,
(2) $H \Vdash \neg\psi$ if $H \nVdash \psi$,
(3) $H \Vdash \psi \to \chi$ if $H \nVdash \psi$ or $H \Vdash \psi$.

The next theorem is the completeness theorem for the hypergraph semantics of our logical system. Note that in Section 7, we stated that the proof of the completeness with respect to the informational semantics requires an introduction of new secrets in addition to those explicitly mentioned in the given formula. Such an extension, however, is not required in the case of the hypergraph semantics.

THEOREM 8.8. *Let V be a set and $\varphi \in \Phi(V)$. If $H \Vdash \varphi$ for each hypergraph H with set V as vertices, then $\vdash \varphi$.*

PROOF. Suppose that $\nvdash \varphi$. Let X be a maximal consistent subset of $\Phi(V)$ containing formula $\neg\varphi$. We define a hypergraph $H = \langle V, E, in, out, w\rangle$ as follows. Let E be set $\{\langle A, p, B\rangle \mid A \rhd_p B \in X\}$. For each edge $\langle A, p, B\rangle \in E$, let $in(\langle A, p, B\rangle) = A$, $out(\langle A, p, B\rangle) = B$, and $w(\langle A, p, B\rangle) = p$. In the four lemmas that follow, we establish basic properties of the hypergraph H needed to finish the proof of the completeness.

LEMMA 8.9. *$in(e) \rhd_{w(e)} out(e) \in X$ for each $e \in E$.*

PROOF. Consider any edge $e = \langle A, p, B\rangle \in E$. Thus, $A \rhd_p B \in X$ by the definition of set E. Hence, $in(e) \rhd_{w(e)} out(e) \in X$ by the definitions of functions in, out, and w. □

LEMMA 8.10. *$X \vdash A \rhd_p A^*_F$ for each finite $F \subseteq E$ such that $w(F) \leq p$ and for each set $A \subseteq V$.*

PROOF. By Lemma 8.5, there must exist $k \geq 0$ such that $A^*_F = A^k_F$. Thus, by Lemma 8.6, there is a sequence

$$A = A_1, f_1, A_2, f_2, \ldots, A_{n-1}, f_{n-1}, A_n = A^*_F \tag{12}$$

such that

(1) $n \geq 1$,
(2) $f_1, \ldots, f_{n-1} \in F$ are distinct edges,
(3) $A_1, \ldots, A_{n-1}$, and A_n are subsets of set V,
(4) $in(f_i) \subseteq A_i$, for each $1 \leq i < n$,
(5) $A_i \cup out(f_i) = A_{i+1}$, for each $1 \leq i < n$.

Towards the proof of the lemma, we first show that

$$X \vdash A \rhd_{\sum_{i=1}^{m-1} w(f_i)} A_m \tag{13}$$

for each $1 \leq m \leq n$. We prove this by induction on m. If $m = 1$, then $A_m = A$ due to the choice of sequence (12). Thus, $X \vdash A \rhd_0 A_1$ by Reflexivity axiom.

Assume that $X \vdash A \rhd_{\sum_{i=1}^{m-1} w(f_i)} A_m$. We need to show that $X \vdash A \rhd_{\sum_{i=1}^{m} w(f_i)} A_{m+1}$. Indeed, since $f_m \in F \subseteq E$, then by Lemma 8.9, we have

$$in(f_m) \rhd_{w(f_m)} out(f_m) \in X.$$

Thus,

$$X \vdash A_m, in(f_m) \rhd_{w(f_m)} A_m, out(f_m)$$

by Augmentation axiom. Note that $in(f_m) \subseteq A_m$ due to the condition 4 above. Hence,

$$X \vdash A_m \rhd_{w(f_m)} A_m, out(f_m).$$

Also note that $A_m \cup out(f_m) = A_{m+1}$ by condition 5 above. Thus,

$$X \vdash A_m \rhd_{w(f_m)} A_{m+1}.$$

Therefore, by the induction hypothesis and Transitivity axiom,

$$X \vdash A \rhd_{\sum_{i=1}^{m} w(f_i)} A_{m+1}.$$

This completes the proof of statement (13). To finish the proof of the lemma, note that the assumption $w(F) \leq p$ implies $\sum_{i=1}^{n-1} w(f_i) \leq p$. At the same time, statement (13) for $m = n$ asserts that $X \vdash A \rhd_{\sum_{i=1}^{n-1} w(f_i)} A_n$. Hence, $X \vdash A \rhd_p A_n$ by Proposition 5.8. Therefore, $X \vdash A \rhd_p A_F^*$ due to the choice of sequence (12). □

LEMMA 8.11. *$A \rhd_p B \in X$ if and only if $H \Vdash A \rhd_p B$, for all sets $A, B \subseteq V$ and each non-negative real number p.*

PROOF. ($\Rightarrow$). Suppose that $A \rhd_p B \in X$. Then, $\langle A, p, B\rangle \in E$ by the choice of set E. Let F be the singleton set $\{\langle A, p, B\rangle\}$. Note that $w(F) = p$, $in(\langle A, p, B\rangle) = A = A_F^0$, and, by Definition 8.3 and Definition 8.4,

$$B = out(\langle A, p, B\rangle) \subseteq \bigcup_{\{f \in F \mid in(f) \subseteq A_F^0\}} out(f) \subseteq A_F^1 \subseteq A_F^*.$$

Hence, $B \subseteq A_F^*$. Therefore, $H \Vdash A \rhd_p B$ by Definition 8.7.
($\Leftarrow$). Suppose that $H \Vdash A \rhd_p B$. Then, by Definition 8.7, there is a finite $F \subseteq E$ such that $w(F) \leq p$ and $B \subseteq A_F^*$. Hence, $\vdash A_F^* \rhd_0 B$ by Reflexivity axiom. Additionally, $X \vdash A \rhd_p A_F^*$ by Lemma 8.10. Thus, $X \vdash A \rhd_p B$ by Transitivity axiom. Therefore, $A \rhd_p B \in X$ due to the maximality of the set X. □

LEMMA 8.12. *$\psi \in X$ if and only if $H \Vdash \psi$ for each $\psi \in \Phi(V)$.*

PROOF. We prove the lemma by induction on the structural complexity of formula ψ. The case when ψ is of form $A \rhd_p B$ follows from Lemma 8.11. The other cases follow from the maximality and consistency of set X and Definition 8.7 in the standard way. □

To conclude the proof of the theorem, note that assumption $\neg\varphi \in X$ implies that $H \nVdash \varphi$ by Lemma 8.12 and Definition 8.7.

9. COMPLETENESS THEOREM FOR THE INFORMATIONAL SEMANTICS

In this section, we prove the completeness theorem for the informational semantics stated previously as Theorem 7.1. This result could be rephrased in terms of statements discussed in the beginning of Section 8 as: the second statement implies the first one. In the previous section, we have already shown that the third statement implies the first one. Thus, it suffices to prove that the second statement implies the third one. In other words, we show that if formula φ is *not* satisfied by a hypergraph H, then it is *not* satisfied by an informational model I_H constructed from H. To prove this, we first describe how to construct an informational model $I_H = \langle \mathcal{A}, \{D_a\}_{a \in \mathcal{A}}, \|\cdot\|, \mathcal{L}\rangle$ based on a given hypergraph $H = \langle V, E, in, out, w\rangle$.

Let $\mathcal{A} = V \cup E$. Recall that sets V and E are disjoint due to Definition 8.1.

Definition 9.1. For any secret $a \in \mathcal{A} = V \cup E$, the cost $\|a\|$ is defined as follows:

$$\|a\| = \begin{cases} w(a), & \text{if } a \in E, \\ +\infty, & \text{if } a \in V. \end{cases}$$

We continue the construction with an auxiliary definition of a path on weighted hypergraph model H. It is convenient to distinguish two types of paths: paths that are initiated at a vertex and paths that are initiated at an edge. Note that paths of both types terminate at a vertex.

Definition 9.2. A path initiated at vertex v_0 is a finite alternating sequence of vertices and edges $\langle v_0, e_1, v_1, e_2, \dots, e_n, v_n \rangle$ such that

(1) $n \geq 0$,
(2) $v_{k-1} \in in(e_k)$ for each $1 \leq k \leq n$,
(3) $v_k \in out(e_k)$ for each $1 \leq k \leq n$.

For example, sequence $\langle v_1, e_1, v_4, e_2, v_6 \rangle$ is a path initiated at vertex v_1 in the hypergraph depicted in Figure 9.

Definition 9.3. A path initiated at edge e_1 is a finite alternating sequence of vertices and edges $\langle e_1, v_1, e_2, \dots, e_n, v_n \rangle$ such that

(1) $n \geq 1$,
(2) $v_{k-1} \in in(e_k)$ for each $1 < k \leq n$,
(3) $v_k \in out(e_k)$ for each $1 \leq k \leq n$.

Sequence $\langle e_1, v_4, e_2, v_6 \rangle$ is a path initiated at edge e_1 in the hypergraph depicted in Figure 9.

To understand the rest of the construction of informational model I_H, let us consider an analogy between this construction and the informal document/folder model discussed in the introduction. The vertices of the hypergraph could be viewed as folders with multiple documents and directed edges show the process of the dissemination and the encryption of these documents between the folders. In our informational model, for the sake of simplicity, each document consists of just a single bit. For the hypergraph depicted in Figure 9, there might be a document (bit) x_{v_6} initially stored in folder v_6. The value of this bit will be disseminated along various paths in the hypergraph and encrypted versions of the document will be stored in different vertices (folders) along these paths. More specifically, the value of each bit stored in a vertex is disseminated *against* the direction of each edge leading to this vertex. In our example, because vertex v_6 is a head of edge e_2 whose tails are v_1 and v_4, the encrypted value of x_{v_6} is disseminated between these two tails. Namely, bit x_{v_6} is represented as a sum of three bits k_{e_2,v_6}, x_{v_1,e_2,v_6} and x_{v_4,e_2,v_6} modulo two:

$$x_{v_6} = k_{e_2,v_6} + x_{v_1,e_2,v_6} + x_{v_4,e_2,v_6} \pmod 2.$$

One can think of bit k_{e_2,v_6} as an encryption key stored in edge e_2 and bits x_{v_1,e_2,v_6} and x_{v_4,e_2,v_6} as encrypted documents distributed between vertices v_1 and v_4. Since vertex v_1 is not a head of any edge in the hypergraph depicted in Figure 9, the value of bit x_{v_1,e_2,v_6} is only stored in vertex v_1 and not disseminated any further. At the same time, because vertex v_4 is a head of edge e_1, the value of bit x_{v_4,e_2,v_6} is further distributed between tails of edge e_1. It is represented as a sum of three bits k_{e_1,v_4,e_2,v_6}, x_{v_1,e_1,v_4,e_2,v_6} and x_{v_2,e_1,v_4,e_2,v_6} modulo two:

$$x_{v_4,e_2,v_6} = k_{e_1,v_4,e_2,v_6} + x_{v_1,e_1,v_4,e_2,v_6} + x_{v_2,e_1,v_4,e_2,v_6} \pmod 2.$$

Again, one can think of bit k_{e_1,v_4,e_2,v_6} as an encryption key stored at edge e_1 and bits x_{v_1,e_1,v_4,e_2,v_6} and x_{v_2,e_1,v_4,e_2,v_6} as encrypted documents distributed between vertices v_1 and v_2.

To summarize, informally, each edge in the hypergraph stores one encryption key corresponding to each path initiated at this edge. Vertices store encrypted documents as they are being disseminated along the paths. This intuition is captured in the two definitions below.

Definition 9.4. For any secret $a \in \mathcal{A} = V \cup E$, let domain D_a be defined as follows:

(1) If $a \in V$, then D_a is the set of all functions that map paths initiated at vertex a into set $\{0, 1\}$.
(2) If $a \in E$, then D_a is the set of all functions that map paths initiated at edge a into set $\{0, 1\}$.

Definition 9.5. Let $\mathcal{L}$ be the set of all vectors $\langle f_a \rangle_{a \in \mathcal{A}} \in \prod_{a \in \mathcal{A}} D_a$ such that for each edge-initiated path $\langle e_1, v_1, e_2, \ldots, e_n, v_n \rangle$, the following equation is satisfied:

$$f_{e_1}(\langle e_1, v_1, e_2, \ldots, e_n, v_n \rangle) + \sum_{u \in in(e_1)} f_u(\langle u, e_1, v_1, e_2, \ldots, e_n, v_n \rangle) = f_{v_1}(\langle v_1, e_2, \ldots, e_n, v_n \rangle) \pmod 2. \tag{14}$$

This concludes the definition of the informational model $I_H = \langle \mathcal{A}, \{D_a\}_{a \in \mathcal{A}}, \| \cdot \|, \mathcal{L} \rangle$.

Definition 9.6. Let $\mathbf{0}$ be the vector $\langle f_a \rangle_{a \in \mathcal{A}}$ such that $f_a(\pi) = 0$ for each $a \in \mathcal{A}$ and each path π initiated at a.

LEMMA 9.7. $\mathbf{0} \in \mathcal{L}$.

PROOF. Equation (14) holds for vector $\mathbf{0}$ because $0 + \sum_{u \in in(e_1)} 0 = 0 \pmod 2$. □

The dissemination of encrypted information described above on the hypergraph depicted in Figure 9 takes place from a head vertex of an edge to the tail vertices of the edge. This means that the bit stored at the head vertex could be determined based on the encryption key and the bits stored at the tail vertices. In other words, information flows in the direction that is opposite to the direction of the edge, but the functional dependency exists in the same direction as the edge. This observation is the key to the understanding of the next lemma.

LEMMA 9.8. *For any set of vertices $A \subseteq V$, any set of edges $F \subseteq E$, any $k \geq 0$, and any two vectors $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{A,F} \ell_2$, then $\ell_1 =_{A^k_F} \ell_2$.*

PROOF. We prove the lemma by induction on k. If $k = 0$, then $A^k_F = A$ by Definition 8.3. Thus, assumption $\ell_1 =_{A,F} \ell_2$ implies that $\ell_1 =_A \ell_2$ by Definition 5.4.

Suppose that $\ell_1 =_{A^k_F} \ell_2$. We need to prove that $\ell_1 =_{A^{k+1}_F} \ell_2$. By Definition 8.3, it suffices to prove that $\ell_1 =_b \ell_2$ for each $e \in F$ and each $b \in out(e)$, where $in(e) \subseteq A^k_F$. See Figure 10.

Let $\ell_1 = \langle f^1_a \rangle_{a \in \mathcal{A}}$ and $\ell_2 = \langle f^2_a \rangle_{a \in \mathcal{A}}$. It suffices to show that

$$f^1_b(\langle b, e_1, v_1, \ldots, e_n, v_n \rangle) = f^2_b(\langle b, e_1, v_1, \ldots, e_n, v_n \rangle)$$

for each path $\langle b, e_1, v_1, \ldots, e_n, v_n \rangle$ initiated at vertex b. Indeed, by Definition 9.5,

$$\begin{aligned} f^1_b(\langle b, e_1, v_1, \ldots, e_n, v_n \rangle) \;=\; & f^1_e(\langle e, b, e_1, v_1, \ldots, e_n, v_n \rangle) + \\ & \sum_{u \in in(e)} f^1_u(\langle u, e, b, e_1, v_1, \ldots, e_n, v_n \rangle) \pmod 2. \end{aligned}$$

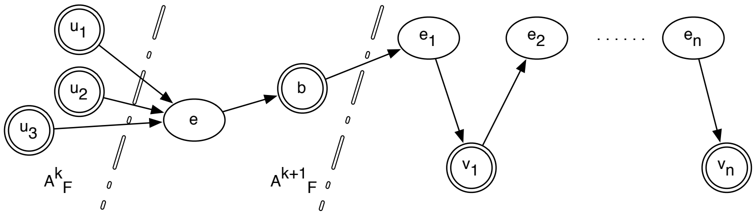

Fig. 10. Illustration to Lemma 9.8.

Recall that $e \in F$, and so, by Definition 5.4, assumption $\ell_1 =_{A,F} \ell_2$ of the lemma implies that $\ell_1 =_e \ell_2$. Hence, $f_e^1 = f_e^2$. Then,

$$\begin{aligned} f_b^1(\langle b, e_1, v_1, \dots, e_n, v_n\rangle) &= f_e^2(\langle e, b, e_1, v_1, \dots, e_n, v_n\rangle) + \\ &\sum_{u\in in(e)} f_u^1(\langle u, e, b, e_1, v_1, \dots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

By induction hypothesis, $\ell_1 =_{A_F^k} \ell_2$. Additionally, $in(e) \subseteq A_F^k$ by the choice of edge e. Thus, $\ell_1 =_{in(e)} \ell_2$ by Definition 5.4. Hence, $f_u^1 = f_u^2$ for each $u \in in(e)$. Then,

$$\begin{aligned} f_b^1(\langle b, e_1, v_1, \dots, e_n, v_n\rangle) &= f_e^2(\langle e, b, e_1, v_1, \dots, e_n, v_n\rangle) + \\ &\sum_{u\in in(e)} f_u^2(\langle u, e, b, e_1, v_1, \dots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

Therefore, $f_b^1(\langle b, e_1, v_1, \dots, e_n, v_n\rangle) = f_b^2(\langle b, e_1, v_1, \dots, e_n, v_n\rangle)$, by Definition 9.5. □

LEMMA 9.9. *For any set of vertices $A \subseteq V$, any set of edges $F \subseteq E$, and any two vectors $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{A,F} \ell_2$, then $\ell_1 =_{A_F^*} \ell_2$.*

PROOF. The statement of the lemma follows from Lemma 9.8 and Lemma 8.5. □

A cut (V_1, V_2) is a partition of the set of all vertices V. We consider cuts to be directed in the sense that (V_1, V_2) and (V_2, V_1) are two different cuts.

Definition 9.10. An edge $e \in E$ is called a crossing edge of a cut (V_1, V_2), if $in(e) \subseteq V_1$ and $out(e) \cap V_2 \neq \varnothing$.

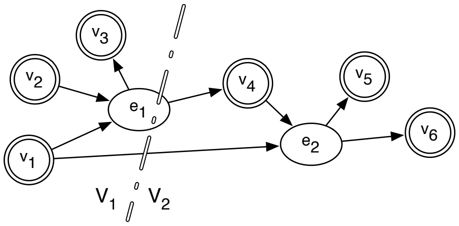

Fig. 11. $Cross(c) = \{e_1\}$, where $c = (\{v_1, v_2, v_3\}, \{v_4, v_5, v_6\})$

The set of all crossing edges of cut $c = (V_1, V_2)$ is denoted by $Cross(c)$. For example, edge e_1 is the only crossing edge of the cut c depicted in Figure 11. Generally speaking, cuts (V_1, V_2) and (V_2, V_1) have different sets of crossing edges.

LEMMA 9.11. *For each $c = (V_1, V_2)$, if $e \notin Cross(c)$ and $out(e) \cap V_2 \neq \varnothing$, then $in(e) \cap V_2 \neq \varnothing$.*

PROOF. Suppose that $in(e) \cap V_2 = \varnothing$. Then, $in(e) \subseteq V_1$. Thus, we have $out(e) \cap V_2 \neq \varnothing$ and $in(e) \subseteq V_1$. Therefore, $e \in Cross(c)$ by Definition 9.10. □

LEMMA 9.12. *If $c = (A^*_F, V \setminus A^*_F)$, then $Cross(c) \cap F = \varnothing$, for each set of vertices $A \subseteq V$ and each set of edges $F \subseteq E$.*

PROOF. Suppose that there is an edge $e \in F$ such that $e \in Cross(c)$. Thus, $in(e) \subseteq A^*_F$ and $out(e) \cap (V \setminus A^*_F) \neq \varnothing$, by Definition 9.10. By Lemma 8.5, there is $k \geq 0$ such that $A^*_F = A^k_F$. Hence, $in(e) \subseteq A^k_F$. Thus, $out(e) \subseteq A^{k+1}_F$ by Definition 8.3. Therefore, by Definition 8.4, $out(e) \subseteq A^*_F$, which yields a contradiction with $out(e) \cap (V \setminus A^*_F) \neq \varnothing$. □

The next definition specifies the core construction in the proof of the completeness by introducing the notion of a *cut-limited inverted tree rooted at a vertex v*. Informally, such a tree starts at vertex v and grows through the edges and vertices of the hypergraph. The tree is *inverted* because it grows in the direction against that of the edges. From each vertex, the tree branches into each edge that has this vertex as a head. From each edge, the tree expands through only one of the tail vertices. Furthermore, if the edge is a crossing edge of the cut, then the tree does not expand from this edge at all. The tree can potentially loop through the hypergraph and be infinite. Figure 12 shows a cut-limited inverted tree rooted at vertex m. Note that this tree is inverted as it expands in the direction opposite to the direction of the edges. The tree is limited by cut (V_1, V_2) and, thus, it does not continue through the crossing edge r of this cut. The tree branches at vertex d and continues through edges r, s, and t of which vertex d is a head. Since the tree does not branch at edges, edge y can only expand either through tail h or through tail k. The inverted tree depicted in Figure 12 expands through tail h.

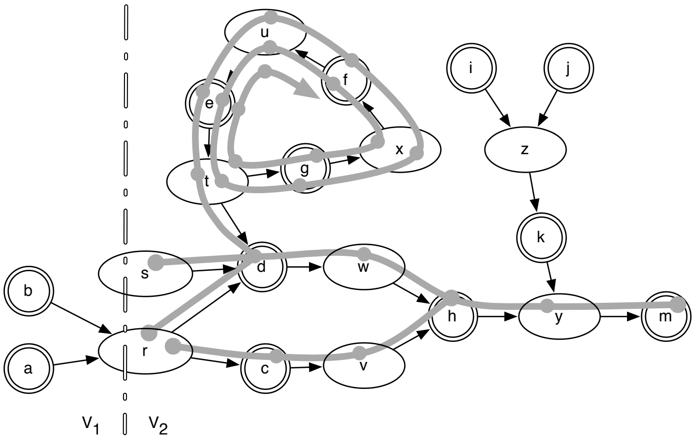

Fig. 12. A cut-limited inverted tree rooted at vertex m.

The next definition specifies the notion of a cut-limited inverted tree. We formally represent tree as a collection of paths.

Definition 9.13. For any cut $c = (V_1, V_2)$, c-limited inverted tree rooted at vertex $v \in V_2$ is any minimal set of paths T such that

(1) $\langle v \rangle \in T$,
(2) for each vertex-initiated path $\langle v_0, e_1, v_1, e_2, \ldots, e_n, v_n \rangle \in T$ and edge $e_0 \in E$ such that $v_0 \in out(e_0)$, we have $\langle e_0, v_0, e_1, v_1, e_2, \ldots, e_n, v_n \rangle \in T$,

(3) for each edge-initiated path $\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle \in T$ if $e_1 \notin Cross(c)$ then there is exactly one $v_0 \in in(e_1) \cap V_2$ such that $\langle v_0, e_1, v_1, e_2, \ldots, e_n, v_n\rangle \in T$.

Because set T in the above definition is required to be a minimal set satisfying the given conditions, all paths in this set terminate with the vertex v at which the tree is "rooted".

LEMMA 9.14. *For any cut $c = (V_1, V_2)$ and any $v \in V_2$ there is a c-limited inverted tree rooted at vertex v.*

PROOF. We recursively construct an infinite sequence of sets of paths $T_0, T_1, T_2, \ldots$, whose vertices are all in set V_2, as follows:

(1) $T_0 = \{\langle v\rangle\}$.
(2) For each $k \geq 0$, let

$$T_{2k+1} = \{\langle e, v_1, e_1, \ldots, v_n, e_n, v\rangle \mid \langle v_1, e_1, \ldots, v_n, e_n, v\rangle \in T_{2k}, v_1 \in out(e)\}.$$

(3) For each path $\pi = \langle e_0, v_1, e_1, \ldots, v_n, e_n, v\rangle \in T_{2k+1}$ such that $e_0 \notin Cross(c)$, choose any vertex $u \in in(e_0) \cap V_2$. Since $v_1 \in out(e_0) \cap V_2$, such vertex u exists by Lemma 9.11. Construct a path $\langle u, e_0, v_1, e_1, \ldots, v_n, e_n, v\rangle$. Let T_{2k+2} be the set of all paths constructed in such a way, taking only one path (and only one vertex u) for each path π.

Let $T = \bigcup_{i\geq 0} T_i$. □

LEMMA 9.15. *For any cut $c = (V_1, V_2)$, any c-limited inverted tree T rooted at vertex $v \in V_2$, and any $\pi \in T$, all vertices in path π belong to set V_2.*

PROOF. The statement of the lemma follows from condition 3 of Definition 9.13 and the minimality condition on T of the same definition. □

The next lemma shows that two vectors can agree on a large set of secrets while not being equal on all secrets.

LEMMA 9.16. *For any vector $\langle f_a\rangle_{a\in\mathcal{A}} \in \mathcal{L}$, any cut $c = (V_1, V_2)$, and any $b \in V_2$, there is a vector $\langle f'_a\rangle_{a\in\mathcal{A}} \in \mathcal{L}$ such that*

(1) $f'_u = f_u$ for each $u \in V_1$,
(2) $f'_e = f_e$ for each $e \in E \setminus Cross(c)$,
(3) $f'_b \neq f_b$.

PROOF. By Lemma 9.14, there exists a c-limited inverted tree T rooted at vertex $b \in V_2$. Define vector $\langle f'_a\rangle_{a\in\mathcal{A}}$ as follows:

$$f'_a(\pi) = \begin{cases} 1 + f_a(\pi), & \text{if } a \in V \text{ and } \pi \in T, \\ 1 + f_a(\pi), & \text{if } a \in Cross(c) \text{ and } \pi \in T, \\ f_a(\pi), & \text{otherwise,} \end{cases} \pmod 2, \tag{15}$$

where π is a path initiated at a. Note that $\pmod 2$ above is a part of $=$ statement and thus applies to all three cases.

Next, we prove that $\langle f'_a\rangle_{a\in\mathcal{A}} \in \mathcal{L}$ and that vector $\langle f'_a\rangle_{a\in\mathcal{A}}$ satisfies conditions (1), (2), and (3) from the statement of the lemma. We state and prove these four facts as Claim 1, Claim 2, Claim 3, and Claim 4 below.

CLAIM 1. *$\langle f'_a\rangle_{a\in\mathcal{A}} \in \mathcal{L}$.*

PROOF. We need to show that vector $\langle f'_a\rangle_{a\in\mathcal{A}}$ satisfies equation (14) of Definition 9.5 for each edge-initiated path. There are three cases that we consider separately:

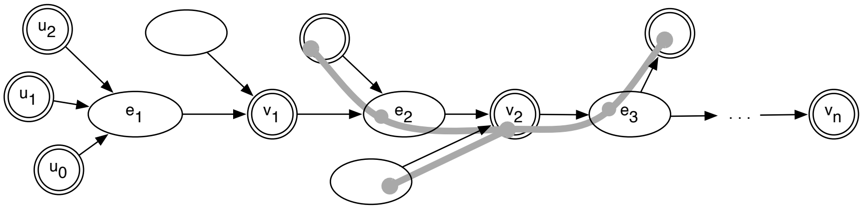

Fig. 13. Case I

Case I: path $\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle$ does not belong to tree T. In this case, by Definition 9.13, path $\langle v_1, e_2, \ldots, e_n, v_n\rangle$ does not belong to tree T either. Neither do paths $\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle$ for each $u \in in(e_1)$. Thus, according to (15),

$$\begin{aligned} &f'_{e_1}(\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f'_u(\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle) \\ &= f_{e_1}(\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f_u(\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

Since $\langle f_a\rangle_{a\in\mathcal{A}} \in \mathcal{L}$, by Definition 9.5,

$$\begin{aligned} &f_{e_1}(\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f_u(\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle) \\ &= f_{v_1}(\langle v_1, e_2, \ldots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

At the same time, since path $\langle v_1, e_2, \ldots, e_n, v_n\rangle$ does not belong to tree T, by (15), we have $f_{v_1}(\langle v_1, e_2, \ldots, e_n, v_n\rangle) = f'_{v_1}(\langle v_1, e_2, \ldots, e_n, v_n\rangle) \pmod 2$. Therefore,

$$\begin{aligned} &f'_{e_1}(\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f'_u(\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle) \\ &= f'_{v_1}(\langle v_1, e_2, \ldots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

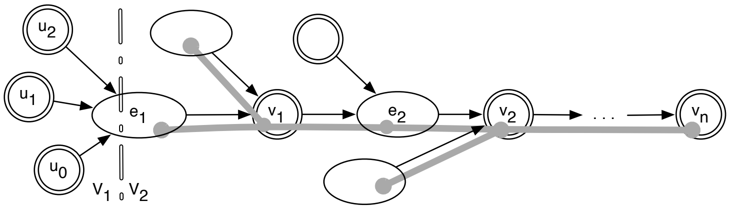

Fig. 14. Case II

Case II: path $\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle$ belongs to tree T and $e_1 \in Cross(c)$. It follows from Definition 9.13 that path $\langle v_1, e_2, \ldots, e_n, v_n\rangle$ belongs to tree T as well and that path $\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle$ does not belong to tree T for each $u \in in(e_1)$. Hence, by (15),

$$\begin{aligned} &f'_{e_1}(\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f'_u(\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle) \\ &= f_{e_1}(\langle e_1, v_1, e_2, \ldots, e_n, v_n\rangle) + 1 + \sum_{u\in in(e_1)} f_u(\langle u, e_1, v_1, e_2, \ldots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

Since $\langle f_a \rangle_{a\in\mathcal{A}} \in \mathcal{L}$, by Definition 9.5,

$$\begin{aligned} & f_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \\ = \ & f_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

At the same time, since path $\langle v_1, e_2, \dots, e_n, v_n\rangle$ belongs to tree T, by (15) we have $f'_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) = f_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) + 1 \pmod 2$. Therefore,

$$\begin{aligned} & f'_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f'_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \\ = \ & f_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) + 1 = f'_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

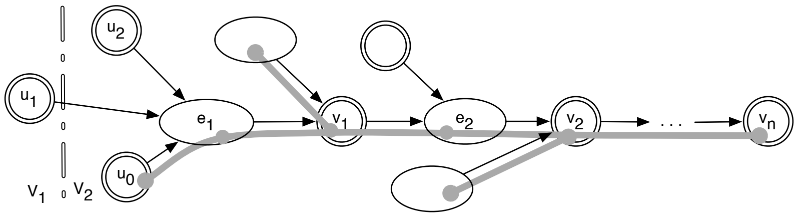

Fig. 15. Case III

Case III: path $\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle$ belongs to tree T and and $e_1 \notin Cross(c)$. It follows from Definition 9.13 that path $\langle v_1, e_2, \dots, e_n, v_n\rangle$ belongs to tree T as well and that there is a unique $u_0 \in in(e_1) \cap V_2$ such that path $\langle u_0, e_1, v_1, e_2, \dots, e_n, v_n\rangle$ belongs to tree T. Hence, by (15),

$$\begin{aligned} & f'_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f'_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \\ = \ & f'_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + f'_{u_0}(\langle u_0, e_1, v_1, e_2, \dots, e_n, v_n\rangle) + \\ & \sum_{u\in in(e_1)\setminus\{u_0\}} f'_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \\ = \ & f_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + (f_{u_0}(\langle u_0, e_1, v_1, e_2, \dots, e_n, v_n\rangle) + 1) + \\ & \sum_{u\in in(e_1)\setminus\{u_0\}} f_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \\ = \ & f_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + 1 + \\ & \sum_{u\in in(e_1)} f_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

Since $\langle f_a \rangle_{a\in\mathcal{A}} \in \mathcal{L}$, by Definition 9.5,

$$\begin{aligned} & f_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + \sum_{u\in in(e_1)} f_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \\ = \ & f_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

At the same time, $f'_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) = f_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) + 1 \pmod 2$ by (15) since path $\langle v_1, e_2, \dots, e_n, v_n\rangle$ belongs to tree T,

$$\begin{aligned} & f'_{e_1}(\langle e_1, v_1, e_2, \dots, e_n, v_n\rangle) + \sum_{u \in in(e_1)} f'_u(\langle u, e_1, v_1, e_2, \dots, e_n, v_n\rangle) \\ = \;& f'_{v_1}(\langle v_1, e_2, \dots, e_n, v_n\rangle) \pmod 2. \end{aligned}$$

This concludes the proof of the claim. □

CLAIM 2. *$f'_u = f_u$ for each $u \in V_1$.*

PROOF. Consider any vertex $u \in V_1$. Recall from Definition 9.4 that the domain of function f_u is the set of all paths starting at vertex u. By Lemma 9.15, none of these paths belongs to tree T. Therefore, $f'_u = f_u$ due to (15). □

CLAIM 3. *$f'_e = f_e$ for each $e \in E \setminus Cross(c)$.*

PROOF. The statement of the claim follows from (15). □

CLAIM 4. *$f'_b \neq f_b$.*

PROOF. By Definition 9.13, the single-element path $\langle b\rangle$ belongs to tree T. Thus, $f'_b(\langle b\rangle) = 1 + f_b(\langle b\rangle) \pmod 2$ due to (15). Therefore, $f'_b \neq f_b$. □

This concludes the proof of the lemma.

LEMMA 9.17. *$H \Vdash A \rhd_p B$ if and only if $I_H \vDash A \rhd_p B$, for each $A, B \subseteq V$ and each non-negative real number p.*

PROOF. ($\Rightarrow$). Suppose that $H \Vdash A \rhd_p B$. Then, by Definition 8.7, there is a subset $F \subseteq E$ such that $w(F) \leq p$ and $B \subseteq A^*_F$. By Definition 9.1, inequality $w(F) \leq p$ implies that $\|F\| \leq p$. Thus, by Definition 5.6, it suffices to show that for any two vectors $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{A,F} \ell_2$, then $\ell_1 =_B \ell_2$, which, in turn, follows from statement $B \subseteq A^*_F$ and Lemma 9.9.
($\Leftarrow$). Assume that $I_H \vDash A \rhd_p B$. Thus, by Definition 5.6, there is $F \subseteq \mathcal{A} = V \cup E$ such that $\|F\| \leq p$ and for all $\ell_1, \ell_2 \in \mathcal{L}$, if $\ell_1 =_{A,F} \ell_2$, then $\ell_1 =_B \ell_2$. Note that, by Definition 9.1, $\|F\| \leq p$ implies that $F \subseteq E$ and $w(F) \leq p$. Suppose now that $H \nVdash A \rhd_p B$. Thus, $B \nsubseteq A^*_F$, by Definition 8.7. Hence, there is $b \in B$ such that $b \notin A^*_F$. To finish the proof of the lemma, it suffices to construct $\ell_1, \ell_2 \in \mathcal{L}$ such that $\ell_1 =_{A,F} \ell_2$ and $\ell_1 \neq_b \ell_2$. Consider cut $c = (A^*_F, V \setminus A^*_F)$. Let ℓ_1 be vector $\mathbf{0} \in \mathcal{L}$. By Lemma 9.16, there is vector ℓ_2 such that $\ell_1 =_{A^*_F, E \setminus Cross(c)} \ell_2$, and $\ell_1 \neq_b \ell_2$. Note that $A \subseteq A^*_F$ by Definition 8.4 and Definition 8.3. Thus, $\ell_1 =_{A, E \setminus Cross(c)} \ell_2$. By Lemma 9.12, we have $Cross(c) \cap F = \varnothing$. In other words, $F \subseteq E \setminus Cross(c)$. Therefore, $\ell_1 =_{A,F} \ell_2$. □

LEMMA 9.18. *$H \Vdash \psi$ if and only if $I_H \vDash \psi$, for each $\psi \in \Phi(V)$.*

PROOF. We prove the lemma by induction on the structural complexity of formula ψ. The base case is shown in Lemma 9.17. The induction step follows from the induction hypothesis, Definition 8.7, and Definition 5.6. □

In the preceding part of this section, given any hypergraph H, we constructed a corresponding informational model I_H and proved properties of this informational model. Next, we finish the proof of the completeness theorem for the informational semantics stated earlier as Theorem 7.1. Assume that $\nvdash \varphi$. By Theorem 8.8, there is a hypergraph $H = \langle V, E, in, out, w\rangle$ such that $\varphi \in \Phi(V)$ and $H \nVdash \varphi$. Therefore, $I_H \nvDash \varphi$ by Lemma 9.18. This concludes the proof of Theorem 7.1.

10. COMPLETENESS THEOREM FOR THE FINITE COST INFORMATIONAL SEMANTICS

In the previous section, we have shown the completeness of our logical system with respect to the informational semantics. In Definition 5.3, we introduced the notion of a finite cost informational model as a model in which each secret has a finite cost. In this section we prove the completeness of our logical system with respect to the class of finite cost models. This is achieved by showing how any (potentially infinite) informational model could be converted to a finite cost model through the construction described below.

Definition 10.1. For any non-negative real number r and any informational model $I = \langle \mathcal{A}, \{D_a\}_{a\in\mathcal{A}}, \|\cdot\|, \mathcal{L}\rangle$, let I^r be tuple $\langle \mathcal{A}, \{D_a\}_{a\in\mathcal{A}}, \|\cdot\|^r, \mathcal{L}\rangle$, where

$$\|c\|^r = \begin{cases} \|c\|, & \text{if } \|c\| \le r,\\ r, & \text{otherwise.}\end{cases}$$

for each secret $c \in \mathcal{A}$.

COROLLARY 10.2. *For any non-negative real number r and any informational model I, tuple I^r is a finite cost informational model.*

COROLLARY 10.3. $\|c\|^r \le \|c\|$*, for each non-negative real number r, each informational model $I = \langle \mathcal{A}, \{D_a\}_{a\in\mathcal{A}}, \|\cdot\|, \mathcal{L}\rangle$, and each secret $c \in \mathcal{A}$.*

Definition 10.4. For any $\varphi \in \Phi(\mathcal{A})$, let $rank(\varphi)$ be defined recursively as follows:

(1) $rank(A \rhd_p B) = p$,
(2) $rank(\neg\psi) = rank(\psi)$,
(3) $rank(\psi \to \chi) = max(rank(\psi), rank(\chi))$.

LEMMA 10.5. *If $rank(\varphi) < r$, then $I^r \vDash \varphi$ if and only if $I \vDash \varphi$.*

PROOF. We prove the lemma by induction on the structural complexity of formula φ. The inductive step immediately follows from Definition 5.6. Now, suppose that formula φ has form $A \rhd_p B$.
($\Rightarrow$) If $I^r \vDash A \rhd_p B$, then, by Definition 5.6, there is a set $C \subseteq \mathcal{A}$ such that (i) $\|C\|^r \le p$ and (ii) for each $\ell_1, \ell_2 \in \mathcal{L}$ if $\ell_1 =_{A,C} \ell_2$, then $\ell_1 =_B \ell_2$. From condition (i), Definition 10.4, and assumption $rank(\varphi) < r$ of the lemma,

$$\|C\|^r \le p = rank(\varphi) < r. \tag{16}$$

Hence, $\|c\|^r \le \|C\|^r < r$ for each $c \in C$, by Definition 5.5. Thus, by Definition 10.1, we have $\|c\| = \|c\|^r$. Then, by Definition 5.5 and the first inequality in statement (16),

$$\|C\| = \sum_{c\in C} \|c\| = \sum_{c\in C} \|c\|^r = \|C\|^r \le p.$$

By Definition 5.6, the inequality $\|C\| \le p$ together with condition (ii) above implies that $I \vDash A \rhd_p B$.
($\Leftarrow$) If $I \vDash A \rhd_p B$, then, by Definition 5.6, there is a set $C \subseteq \mathcal{A}$ such that (iii) $\|C\| \le p$ and (iv) for each $\ell_1, \ell_2 \in \mathcal{L}$ if $\ell_1 =_{A,C} \ell_2$, then $\ell_1 =_B \ell_2$. By Definition 5.5, Corollary 10.3, and again Definition 5.5,

$$\|C\|^r = \sum_{c\in C} \|c\|^r \le \sum_{c\in C} \|c\| = \|C\|.$$

This along with condition (iii) implies that $\|C\|^r \le p$. Therefore, by Definition 5.6 and condition (iv), we have $I^r \vDash A \rhd_p B$. □

We next state and prove the completeness theorem for the class of finite cost informational models.

THEOREM 10.6. *If $I \vDash \varphi$ for each finite cost informational model $I = \langle \mathcal{A}, \{D_a\}_{a\in\mathcal{A}}, \|\cdot\|, \mathcal{L}\rangle$ such that $\varphi \in \Phi(\mathcal{A})$, then $\vdash \varphi$.*

PROOF. Suppose that $\nvdash \varphi$. Thus, by Theorem 7.1, there is an informational model $I = \langle \mathcal{A}, \{D_a\}_{a\in\mathcal{A}}, \|\cdot\|, \mathcal{L}\rangle$ such that $I \nvDash \varphi$. Pick any r such that $rank(\varphi) < r$. Then, $I^r \nvDash \varphi$ by Lemma 10.5. □

11. CONCLUSION

In this article we have introduced a notion of budget-constrained dependency that generalizes the notion of functional dependency previously studied by Armstrong [1974]. We propose a sound and complete axiomatization that captures the properties of the budget-constrained dependency. Although the axioms of our system are generalizations of Armstrong's original axioms, the proof of the completeness for our system is significantly more complicated than Armstrong's counterpart.

REFERENCES

Natasha Alechina and Brian Logan. 2002. Ascribing beliefs to resource bounded agents. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2002)*, Vol. 2. ACM Press, Bologna, 881–888.

Natasha Alechina, Brian Logan, Hoang Nga Nguyen, and Abdur Rakib. 2011. Logic for coalitions with bounded resources. *Journal of Logic and Computation* 21, 6 (December 2011), 907–937.

Natasha Alechina, Brian Logan, and Mark Whitsey. 2004. A Complete and Decidable Logic for Resource-Bounded Agents. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004)*, Nicholas R. Jennings, Charles Sierra, Liz Sonenberg, and Milind Tambe (Eds.). ACM Press, New York, 606–613.

W. W. Armstrong. 1974. Dependency structures of data base relationships. In *Information Processing 74 (Proc. IFIP Congress, Stockholm, 1974)*. North-Holland, Amsterdam, 580–583.

Catriel Beeri, Ronald Fagin, and John H. Howard. 1977. A complete axiomatization for functional and multivalued dependencies in database relations. In *SIGMOD '77: Proceedings of the 1977 ACM SIGMOD international conference on Management of data*. ACM, New York, NY, USA, 47–61. DOI: http://dx.doi.org/10.1145/509404.509414

Radim Bělohlávek and Vilém Vychodil. 2006. Data tables with similarity relations: functional dependencies, complete rules and non-redundant bases. In *Database Systems for Advanced Applications*. Springer, 644–658.

Claude Berge. 1989. *Hypergraphs*. North-Holland Mathematical Library, Vol. 45. North-Holland Publishing Co., Amsterdam. x+255 pages. Combinatorics of finite sets, Translated from the French.

Nils Bulling and Berndt Farwer. 2010. Expressing properties of resource-bounded systems: The logics RTL* and RTL. In *Computational Logic in Multi-Agent Systems*. Springer, 22–45.

Hector Garcia-Molina, Jeffrey Ullman, and Jennifer Widom. 2009. *Database Systems: The Complete Book* (second ed.). Prentice-Hall.

Jean-Yves Girard. 1987. Linear Logic. *Theoretical computer science* 50 (1987), 1–102.

S. Hartmann, S. Link, and K.D. Schewe. 2004. Weak functional dependencies in higher-order datamodels. In *International Symposium on Foundations of Information and Knowledge Systems*. Springer, Berlin Heidelberg, 116–133.

Zachary Heckle and Pavel Naumov. 2014. Common Knowledge Semantics of Armstrong's Axioms. In *Proceedings of 21st Workshop on Logic, Language, Information and Computation (WoLLIC), September 1st to 4th, 2014, Valparaiso, Chile*. Springer, 181–194.

Wojciech Jamroga and Masoud Tabatabaei. 2013. Accumulative Knowledge under Bounded Resources. In *Computational Logic in Multi-Agent Systems*, João Leite, Tran Cao Son, Paolo Torroni, Leon van der Torre, and Stefan Woltran (Eds.). Lecture Notes in Computer Science, Vol. 8143. Springer Berlin Heidelberg, 206–222.

Jonathan Katz. 2010. *Digital Signatures*. Springer Science & Business Media.

Pavel Naumov and Brittany Nicholls. 2014. Rationally Functional Dependence. *Journal of Philosophical Logic* 43, 2-3 (2014), 603–616.

Pavel Naumov and Jia Tao. 2015. Budget-Constrained Knowledge in Multiagent Systems. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. International Foundation for Autonomous Agents and Multiagent Systems, 219–226.

Pavel Naumov and Jia Tao. 2016a. Information Flow Under Budget Constraints. In *European Conference on Logics in Artificial Intelligence (JELIA)*. Springer, 353–368.

Pavel Naumov and Jia Tao. 2016b. Price of Privacy. In *12th Conference on Logic and the Foundations of Game and Decision Theory (LOFT), Maastricht, the Netherlands*.

Jouko Väänänen. 2007. *Dependence logic: A new approach to independence friendly logic*. Vol. 70. Cambridge University Press.

Jouko Väänänen. 2017. The logic of approximate dependence. In *Rohit Parikh on Logic, Language and Society*. Springer, 227–234.