Universal Semantic Communication I

Brendan Juba* Madhu Sudan †

MIT CSAIL
{bjuba,madhu}@mit.edu

September 3, 2007

Abstract

Is it possible for two intelligent beings to communicate meaningfully, without any common language or background? This question has interest on its own, but is especially relevant in the context of modern computational infrastructures where an increase in the diversity of computers is making the task of inter-computer interaction increasingly burdensome. Computers spend a substantial amount of time updating their software to increase their knowledge of other computing devices. In turn, for any pair of communicating devices, one has to design software that enables the two to talk to each other. Is it possible instead to let the two computing entities use their intelligence (universality as computers) to learn each others' behavior and attain a common understanding? What is "common understanding?" We explore this question in this paper.

To formalize this problem, we suggest that one should study the "goal of communication:" why are the two entities interacting with each other, and what do they hope to gain by it? We propose that by considering this question explicitly, one can make progress on the question of universal communication.

We start by considering a computational setting for the problem where the goal of one of the interacting players is to gain some computational wisdom from the other player. We show that if the second player is "sufficiently" helpful and powerful, then the first player can gain significant computational power (deciding PSPACE complete languages).

Our work highlights some of the definitional issues underlying the task of formalizing universal communication, but also suggests some interesting phenomena and highlights potential tools that may be used for such communication.

*Supported by an Akamai Presidential Fellowship, a NSF Graduate Research Fellowship, and NSF Award CCR-0514915.

†Supported in part by NSF Award CCR-0514915.

1 Introduction

Consider the following scenario: Alice, an extraterrestrial, decides to initiate contact with a terrestrial named Bob by means of a radio wave transmission. How should he respond to her? Will he ever be able to understand her message? In this paper we explore such scenarios by framing the underlying questions computationally.

We believe that the above questions have intrinsic interest, as they raise some further fundamental questions. How does one formalize the concept of understanding? Does communication between intelligent beings require a "hardwired" common sense of *meaning* or *language*? Or, can intelligence substitute for such requirements? What role, if any, does computational complexity play in all this? We also examine some practical motivations for studying this topic at length in Section 3.1.

1.1 Some history

Despite the fantastic nature of the topic (of conversation with an alien), it has been seriously considered in the past, though mostly in an empirical context. Most notably, Freudenthal [2] claims that it is possible to code messages describing mathematics, physics, or even simple stories in such a radio transmission which can be understood by any sufficiently humanlike recipient. Ideally, we would like to have such a rich language at our disposal; it should be clear that the "catch" lies in Freudenthal's assumption of a "humanlike" recipient, which serves as a catch-all for the various assumptions that serve as the foundations for Freudenthal's scheme.

It is possible to state more precise assumptions which form the basis of Freudenthal's scheme, but among these will be some fairly strong assumptions about how the recipient interprets the message. In particular, one of these is the assumption that all semantic concepts of interest can be characterized by lists of syntactic examples. (See Appendix A for a more detailed discussion.) These assumptions seem, for the moment, to be too strong to be considered truly universal.

In our work, we attempt to reduce the "commonality" assumptions to more semantic ones. To illustrate what we mean by this, suppose that the two communicating agents share knowledge of the laws of physics, or mathematical principles (they may have the same system of logic, and rely on some form of Peano's axioms etc.). If so, we would prefer not to suggest a specific form in which they store this information, but rather are aware of what is feasible or infeasible under such laws. Of course, our assumptions will be essentially about common awareness of information-theoretic and computational "laws."

Communication has, of course, been studied from a mathematical perspective before, and so we mention the difference between the goal there and in our case. The classical theory of communication [10] does not investigate the meaning associated with information and simply studies the process of communicating the information, in its exact syntactic form. It is the success of this theory that motivates our work: computers are so successful in communicating a sequence of bits, that the most likely source of "miscommunication" is a misinterpretation of what these bits mean.

The theory of communication complexity [11] starts to associate meaning to the bits being communicated e.g., the reason Bob wants Alice to send x is that he wants to compute some function $f(x, y)$ where he knows y and both Alice and Bob know $f(\cdot, \cdot)$. So the "intent" of the communication is to enable Bob to compute $f(x, y)$ and knowing this intent, one may be able to achieve further efficiency in the communication. The idea that Bob may wish to know x for a given reason, will be useful in our setting as well, but the common knowledge of "f" does not apply to our setting.

Finally, the theory of interactive proofs and knowledge [5] (and also the related theory of program checking [1]) gets further into the gap between Alice and Bob, by ascribing to them different intents,

though they still share common semantics. It turns out this gap already starts to get to the heart of the issues that we consider, and this theory is very useful to us at a technical level. In particular, in this work we consider a setting where Bob wishes to gain knowledge from Alice. Of course, in our setting Bob is not mistrustful of Alice, he simply does not understand her.

1.2 Modeling issues

Our goal is to cast the problem of "meaningful" communication between Alice and Bob in a purely mathematical setting. We start by considering how to formulate the problem where the presence of a "trusted third party" would easily solve the problem.

Consider the informal setting in which Alice and Bob speak different natural languages and wish to have a discussion via some binary channel. We would expect that a third party who knows both languages could give finite encoding rules to Alice and Bob to facilitate this discussion, and we might be tempted to require that Alice's statements translate into the same statements in Bob's language that the third party would have selected and vice-versa.

In the absence of the third party, this is unreasonable to expect, though: suppose that Alice and Bob were given encoding rules that were identical to those that a third party would have given them, except that some symmetric sets of words have been exchanged—say, Alice thinks "left" means "right," "clockwise" means "counter-clockwise," etc. Unless they have some way to tell that these basic concepts have been switched, observe that they would still have a conversation that is entirely sensible to each of them.[1] Thus, if we are to have any hope at all, we must be prepared to accept interactions that are indistinguishable from successes as "successes" as well. We do not wish to take this to an extreme, though: Bob cannot distinguish among Alices who say nothing, and yet we would not classify their interactions as "successes."

At the heart of the issues raised by the discussion above is the question: What does Bob hope to get out of this conversation with Alice? In general, why do computers, or humans communicate? Only by pinning down this issue can we ask the question, "Can they do it without a common language?"

1.3 Our approach and results

To see how our search for "universal communication" leads to defininig "goals of communication," we start with a simple computational motivation for communication (from Bob's perspective).

Consider the case where Bob is a probabilistic polynomial time bounded interactive machine, whose goal is to decide membership in some set S. (For instance, Bob's goal may be to detect if a given program is a virus and so the set S may be the set of programs that are not viruses.) His hope is that by interacting with Alice, he can solve this hard computational problem. We consider an all-powerful Alice who can decide membership in any set. If Alice and Bob understood and trusted each other completely, then this could allow Bob to solve any membership problem! Note that the setting where Alice and Bob understand each other, but don't necessarily trust each other is the setting considered in interactive proofs, and here we have that Bob can (only) solve decision problems in PSPACE [7, 9]. Our setting is the "dual" one where Alice and Bob don't necessarily mistrust each other, but they definitely don't start off understanding each other. Somewhat strikingly, we show that this setting turns into the same setting as its dual, and that Bob can decide problems (only) in PSPACE.

[1] Bob Berwick has pointed out to us that Quine [8] makes a similar but much stronger assertion of *indeterminacy of translation*. Frustratingly, he leaves justification for these claims largely to the reader's imagination.

We show this in two parts: First we show that given an all-powerful (or even a PSPACE-powerful) and sufficiently helpful Alice, Bob can decide membership for any problem in PSPACE (see Theorem 4). So, Alice manages to communicate the answer to any set membership question to Bob in spite of their lack of a common language to begin with. The crux of this step is to come up with a proper definition of "sufficiently helpful" that helps avoid pitfalls such as expecting to do this with an Alice that doesn't communicate, or just spits out random answers, yet also without restricting Alice so much that the issue of common language simply goes away. Our definition is given in Section 2.2 where we discuss why this is a ("most") reasonable definition. Next, we show that under any sufficiently strong definition of "lack of common language" between Alice and Bob, Bob can not solve problems outside PSPACE even if Alice is sufficiently helpful (see Theorem 6).

Both our proofs above are based on "enumeration" arguments. In the case where $S \in$ PSPACE, Bob enumerates potential methods for intepreting Alice's language and rules out incorrect choices by using his power to verify answers given by Alice (using this interpreted version of Alice as a prover). This argument extends a similar argument by Levin [6] giving an optimal algorithm for NP-search problems, assuming NP=P. In fact, as pointed out by Oded Goldreich [3], our argument can be abstracted as giving an (explicit) randomized polynomial time algorithm for PSPACE assuming PSPACE=BPP. We remark that positive uses of enumeration are somewhat rare in complexity (as opposed to the standard use for proving impossibility results) and we are only aware of such use in [6] and in [4].

Our limitation result, showing S must be in PSPACE, considers some S outside PSPACE and uses diagonalization to show that for every Bob there exists an Alice that is helpful (to somebody) and yet communicates misleading things to Bob.

We remark that our proof only yields a Bob that errs finitely many times in interacting with Alice. Note that in the setting of universal communication, the presence of a "universal Bob" that makes only finitely many mistakes does not imply the presence of another "universal Bob" that makes no mistakes. Indeed this weakness of our (negative) result may be for inherent reasons: if Bob is allowed to err finitely many times, then he might be able to use a helpful Alice to solve any decidable problem. We believe such a result holds in some variants of the model presented here, and this will be addressed in further work.

The enumeration argument in our positive result yields a somewhat weak positive result. In order to establish communication between Alice and Bob, Bob runs in time exponential in a parameter that could be described informally as the length of the dictionary that translates Bob's language into Alice's language. (Formally, the parameter is the description length of the protocol for interpreting Alice in his encoding of Turing machines.) In Theorem 7, we argue that such an exponential behavior is necessitated by our model of "lack of understanding" assuming BPP $\neq$ PSPACE, and a sufficiently general class of "Alices."

2 Communication with a Computational Goal

In this section we formalize a computational goal for communication and prove feasibility and infeasibility results for universal communication in this setting. More specifically, we define a notion of a "helpful" Alice, and the goal of a "universal" Bob. We show that there is a universal Bob that can decide PSPACE complete problems when interacting with any sufficiently powerful and helpful Alice. We also prove a matching negative result showing that Bob can not decide problems outside of PSPACE, provided the "language" of Alice is sufficiently unknown. Again crucial to this step is formalizing the concept of the language being unknown.

2.1 Basic notation

We start by setting up our basic notation for interactive computation (borrowed from the classical setting of interactive proofs). We assume that Alice and Bob exchange messages by writing finite length strings from a binary alphabet on common tape. We assume they are well synchronized[2], i.e., they alternate writing, and they know when the other has finished writing. Thus a history $\mathbf{m}$ of the interaction consists of a sequence of strings $\mathbf{m} = \langle m_1, \ldots, m_k \rangle$ where $m_i \in \{0,1\}^*$. The odd messages are written by Alice, and the even ones by Bob. Each player may toss random coins and have some private inputs. For example, if the $k+1$th message is written by Alice, it is a function of the history thus far, as well as Alice's randomness and private inputs. We describe her response by the function $\mathrm{A}(\mathbf{m})$. (We remark that we don't highlight her randomness and private inputs, since these will be irrelevant to Bob.) Similarly Bob's messages are also (probabilistic) functions of the history and any private inputs he may have. Bob's message on private input x and history $\mathbf{m}$ is denoted $\mathrm{B}(x;\mathbf{m})$. Once again this function may depend on the history of random coins tossed by Bob but we will suppress this aspect in our notation.

Conditioned on a history $\mathbf{m}$, Alice's responses in the future may be viewed as a new incarnation of Alice. We use $\mathrm{A}_{\mathbf{m}}$ to denote her future responses and thus $\mathrm{A}_{\mathbf{m}}(\mathbf{m}') = \mathrm{A}(\mathbf{m} \circ \mathbf{m}')$ where $\mathbf{m} \circ \mathbf{m}'$ denotes the concatenation of the histories $\mathbf{m}$ and $\mathbf{m}'$.

At the end of an interaction with Alice, Bob will output a Boolean verdict. $(\mathrm{A}, \mathrm{B}(x))$ will denote the random variable produced by Bob's output following the interaction between Alice and Bob, where Bob has private input x. We will abuse notation in a natural way to let a decision problem L also denote a Boolean function: $L(x) = 1$ if $x \in L$ and $L(x) = 0$ otherwise.

Definition 1 *We say that* Alice helps Bob decide a problem L *if for every* $x \in \{0,1\}^*$, *it is the case that* $\Pr\left[(\mathrm{A}, \mathrm{B}(x)) = L(x)\right] \geq 2/3$.

2.2 Main Definitions and Results

Our general approach is to consider the situation where Bob interacts with some member of a large class $\mathcal{A}$ of Alices, but does not know which specific member of the class he is interacting with. Essentially, we would like Bob to be successful in deciding the problem L for every member of the class $\mathcal{A}$. (Sometimes we may also wish to consider what Bob does when Alice does not belong to $\mathcal{A}$.) In order to make this viable, the class $\mathcal{A}$ should only include Alices that are powerful (enough to decide L), and helpful. While the former is easy to formalize, the latter notion is somewhat subtle. One aspect we'd like to capture here is that her ability to decide L should be based on her "external characteristics," namely on her input/output response, but this is still insufficient. For instance suppose that for each round i, Alice has chosen a random bit b_i (known only to her) and then answers any question y with the bit $L(y) \oplus b_i$. In this setting her input/output response at time i represents her ability to decide L – someone who knew some b_i would be able to easily obtain $L(y)$ for any y – but this is clearly not detectable by (poor) Bob. This introduces an additional element that "helpfulness" must capture, namely Alice's behavior as a function of the history of messages thus far.

In the following definition we attempt to formalize the notion of a powerful and helpful Alice by setting minimal restrictions on Alice. Roughly, we insist that Alice be able to help *some* Bob$'$ decide L, conditioned on *any* prior history. The requirement that Alice should be able to help

[2]This and similar syntactic assumptions may already be questioned in the general setting of "intergalactic communication." Indeed these simplify our task, however they do not trivialize the problem and we are optimistic that these can be removed at a later stage.

some Bob$'$ decide L is necessary if we would like to design a specific Bob to decide L by interacting with Alice. Moreover, observe that if no such Bob$'$ exists, no matter what is said to Alice and no matter in what language, Alice provides no assistance, so the difficulty is surely not merely one of "lack of a common language." The requirement that this should happen independent of any history does restrict Alice somewhat, but we argue that it is a simple way to overcome issues such as the time-varying Alice described above, and therefore a reasonable "universal" principle.

Definition 2 (L-Helpful Alice) *We say that Alice is* L-helpful *if there exists a probabilisitic polynomial time bounded Bob, such that for every prior history* $\mathbf{m}$*, the incarnation of Alice conditioned on the history,* $A_{\mathbf{m}}$*, helps Bob decide* L*.*

We now formalize Bob's goal in this computational setting.

Definition 3 (L-Universal) *We say that Bob is a universal decider for a decision problem* L*, or* L*-universal, if for any* L*-Helpful Alice, Alice helps Bob decide* L *and there exists a polynomial* p *such that for every instance* $x \in \{0,1\}^*$ *Bob runs in expected time* $p(|x|)$*.*

Our main theorem is the following result, which gives a universal Bob for problems in PSPACE.

Theorem 4 *For every PSPACE complete problem* L*, there is a Bob that is* L*-universal.*

This result uses the power of interactive proofs and in particular the fact that PSPACE has short interactive proofs [7, 9]. Effectively Bob attempts to get from Alice, not only the answer to a question of the form "Is $x \in L$?", but also an interactive proof of this answer.

Next we will rule out the possibility of obtaining a L-Universal Bob for $L \notin PSPACE$. We prove this by showing that if we consider any sufficiently broad class of helpful Alices $\mathcal{A}$, then for every probabilistic polynomial time bounded Bob, there exists an Alice in the class $\mathcal{A}$ such that the decision problem (promise problem to be more precise) that Alice helps Bob decide is contained in PSPACE. Of course this theorem in turn relies on the definition of "sufficient broadness" above, which we specify next.

Definition 5 (Semantically closed class) *We say that a class of Alices* $\mathcal{A}$ *is semantically closed if for every injective function* f *where* f *and* f^{-1} *are computable in polynomial time, and for every member* $A \in \mathcal{A}$*, it is the case that every incarnation* A^f *satisfying* $A^f(f(m_1), \ldots, f(m_k)) = A(m_1, \ldots, m_k)$*, is also in* $\mathcal{A}$*.*

An incarnation A^f is simply A who uses the dictionary f to convert from A's language to hers and the dictionary f^{-1} to convert back. Note that A^f may have arbitrary behavior outside the range of f! We now state our impossibility result for L-universal deciders for $L \notin$ PSPACE.

Theorem 6 *Let* L *be a decision problem that is not in PSPACE. Let* $\mathcal{A}$ *be a semantically closed class of* L*-helpful Alices. Then for every probabilistic algorithm* B*, there exists an Alice* $A \in \mathcal{A}$ *such that* B *fails to decide* L *with the help of* A*.*

The insistence that we deal only with semantically closed classes is somewhat broad so we discuss the definition next. This definition is consistent with our goal that we make few assumptions about language, and this definition suggests that every polynomial time computable translation of a "natural language" is also natural. This is, of course, somewhat broad, yet we don't believe this

is the aspect that limits the power of universal semantics. This is merely a definition under which we can *prove* the limitations.

Perhaps a more distressing consequence of the level of generality we seek is that the running time of the universal Bob constructed in Theorem 4 is exponentially long in the description length of the shortest asymptotically optimal protocol for interpreting Alice (in Bob's encoding). Notice that with a trusted third party, Bob would have had only a polynomial dependence on the encoding of this protocol. The following theorem asserts that this is necessary:

Theorem 7 *Let L be a PSPACE-complete decision problem, let $\mathcal{A}$ be a semantically closed class of L-helpful Alices, and let $\mathcal{A}|_t \subset \mathcal{A}$ be the subclass that help some protocol running in time $O(t(n))$. Then, unless PSPACE=BPP, if a probabilistic algorithim Bob decides instances of L using the help of an arbitrary Alice in $\mathcal{A}|_t$ in time $O(n^k)$, Bob's running time must also be exponential in the description length of the shortest protocol that is helped by Alice in time $O(t(n))$.*

To prove this theorem we exhibit an infinite family of Alices such that each Alice helping a protocol of description length $k+O(1)$ expects a different "password" of length k. Since Alice is a black-box to Bob, a Bob running in time subexponential in k does not have time to try every password, and thus must output a verdict without Alice's help.

2.3 Proofs of theorems

We begin by observing that it is a trivial consequence of the definition of L-Universal that any L-Universal Bob satisfies the following "weak soundness" condition:

Definition 8 (Weak Soundness) *We say that Bob has* weak soundness *if for every L-helpful Alice and all x, the probability that $(A, B(x)) \neq L(x)$ is at most $1/3$.*

By contrast, we could have considered the following "strong" definition of L-Universality

Definition 9 (Strongly L-Universal) *We say that Bob is a strongly universal decider for a decision problem L, or* strongly L-universal, *if it satisfies the following conditions:*

Completeness *If Alice is L-Helpful, then Alice helps Bob decide L and there exists a polynomial p such that for every $x \in \{0,1\}^*$ Bob runs in expected time $p(|x|)$.*

Soundness *For every Alice and all x, the probability that $(A, B(x)) \neq L(x)$ is at most $1/3$.*

Before proving our main theorems in this setting, we discuss this definition a bit, especially the strong vs. weak soundness condition. We are envisioning an exchange between a helpful Alice and Bob, and the completeness condition merely states our goal that Bob should be able to exploit Alice's help. The soundness condition leads to a choice. Bob doesn't understand Alice; should he trust her in spite of this? The weak soundness condition is the trusting scenario, and says that Bob should not make incorrect decisions when Alice is being helpful, though if Alice is malicious/non-helpful, there are no guarantees. The strong soundness condition is the more xenophobic one, allowing the lack of understanding to lead to a lack of trust. As we will see below, the distinction turns out to be really not important.

For our main theorem, we will actually prove the following (seemingly stronger) positive result, which gives a strongly universal Bob for PSPACE-complete problems.

Theorem 10 (Theorem 4, strong version) *For every PSPACE complete problem L, there is a Bob that is strongly L-universal.*

Proof (of Theorem 4): Recall that we need to construct a Bob B that can interact with an Alice A to compute $L(x)$. We start with an overview first.

Note that since Alice is helpful there exists some Bob B^* that can decide L with Alice's help. The Bob B we construct (using an idea originally due to Levin [6] and further developed by Goldreich and Ron [4]) enumerates all Bobs $\tilde{B}$ hoping to guess B^*. If $\tilde{B} = B^*$ then Bob gets an "oracle prover" with the power to answer any PSPACE problem. This allows our Bob B to simulate an interactive proof of the fact that $x \in L$ (using the fact that PSPACE in contained in IP [7, 9], where the prover only needs be in PSPACE). If $\tilde{B} \neq B^*$, then since B can never be convinced of a wrong assertion (of the form $x \in L$ when actually $x \notin L$), this simulation does not lead B astray. It simply rules out $\tilde{B}$ as the guess and moves on to the next guess for B^*. In the details below we attempt to ensure that B's running time is not too large when Alice is helpful, and also to ensure termination (though not in polynomial time) when Alice is not helpful.

Construction of B: Let V_L denote the verifier for the language $L' = \{(x, b) | L(x) = b\}$. We assume that V_L exchanges at most $r(|x|)$ messages with any given prover for some polynomial r. Let P_L denote the "optimal binary prover" for this language, i.e., $P_L(x, b, \mathbf{m}, y)$ gives the Boolean answer to question y that is most likely to cause V_L to accept on input (x, b) after history of interactions being $\mathbf{m}$. Note the P_L is computable in PSPACE and so there exists a reduction f_L such that f_L reduces the optimal prover's computation to a decision problem in L. In other words $P_L(x, b, \mathbf{m}, y) = L(f_L(x, b, \mathbf{m}, y))$. We show how to use f_L and V_L to help construct Bob B. Our final Bob B runs two Bobs B_1 and B_2 in parallel and halts whenever either of them halts and outputs the answer of the one that halted.

B_1 is simply the Bob that runs in time $2^{p_1(|x|)}$ for some polynomial p_1 to decide if $L(x) = b$. We hope not to use this Bob much with a helpful Alice. It is used to provide a bound on the number of times we simulate the interaction between P_L and V_L, and thereby permits us to guarantee a negligible probability of failure.

B_2 is the crucial element in our construction. On input x, B_2 enumerates triples $(\tilde{B}, t, b)$ where $\tilde{B}$ is a probabilistic algorithm, t is an integer, and b is a bit. Following Goldreich and Ron [4] we enumerate triples $(\tilde{B}, t, b)$ in phases: in the ith phase, for both $b \in \{0, 1\}$, we enumerate all $\tilde{B}$ having programs of length at most $i - 2 \lg i$, with each program of length ℓ having $t = \frac{2^i}{\ell^2 2^\ell}$. For each such triple, B_2 attempts to use $(A, \tilde{B}(\cdot))$ as an oracle for PSPACE to simulate the interactive proof to verify if $L(x) = b$ as follows:

- It simulates the probabilistic computation of the verifier V_L.
- To simulate the optimal prover it attempts to compute $P_L(x, b, \mathbf{m}, y) = L(f_L(x, b, \mathbf{m}, y))$ by interacting with A according to $\tilde{B}(x, b, \mathbf{m}, y)$, and taking the verdict of $\tilde{B}$ as the prover's message.

For $(\tilde{B}, t, b)$, in total we only simulate up to t steps of these computations. We repeat this simulation $n + 18p_1(n)$ times, and we keep count of how many times the conversation halts with V_L accepting. If the majority of the simulations accept, then B_2 halts and outputs $L(x) = b$; if not it continues on to the next triple $(\tilde{B}, t, b)$.

Analysis: We now analyze the completeness and soundness of B. The soundness of B is a straightforward consequence of the soundness of the verifier V_L, the verifier for the language L';

notice that we finish within $2^{p_1(n)}$ phases, where we have reduced the probability of failure in a phase to $2^{-p_1(n)} \cdot \mathrm{negl}(n)$. Thus, if V_L halts and accepts a pair (x, b) with noticeable probability, it must be that $L(x) = b$. So this yields that the probability that B_2 halts and produces the wrong answer is negligible. B_1 never produces the wrong answer so we always have the right answer.

To finish up, we need to show that if A is L-helpful then B halts in some fixed polynomial time and outputs $L(x)$. To see this note that B^*, the Bob that makes A helpful, must be enumerated sometime by B_2. Since B^* is efficient, its computation can be simulated in $p_2(n)$ steps for some polynomial p_2. Likewise, the computation of V_L can be simulated in at most $p_3(n)$ steps for some polynomial p_3, and thus if $t \geq r(n)(p_2(n)+p_3(n))$, the simulation runs to completion in t steps. At this stage B_2 has access to an optimal prover for PSPACE which will convince it to accept $L(x) = b$, for the right choice of b. This ensures completeness of the Bob B. It is easy to verify that if B^* has length ℓ^*, the choice of enumeration guarantees that B_2 spends at most $2^{O(\ell^*)}t(n + 18p_1(n))$ steps before enumerating the triple $(\tilde{B}, t, b)$. Since the simulations complete for $t \geq r(n)(p_2(n) + p_3(n))$, B runs for at most $O\big(r(n)(p_2(n) + p_3(n))(n + p_1(n))\big)$ steps in total whenever A is helpful. ■

The one weakness in our definitions and constructions is that we allow Bob's running time to depend on Alice. This is essential to get universality and we prove this formally in Proposition 11 below.

Proposition 11 *If, for a PSPACE complete problem L, there exists a B and a k such that for every L-helpful A, B decides L with the A's help in time $O(n^k)$, then $BPP = PSPACE$.*

Proof The proof is a simple padding argument. Assume that such a Bob B exists with running time $O(n^k)$ (that achieves exponentially small error). Consider an Alice who decides $\{x10^{|x|^{2k}} : x \in L\}$. Clearly such an Alice is helpful. Now consider the task of deciding if $x \in L$ when $n = |x|$ is sufficiently large. While trying to decide this, Bob only has time to query Alice with queries of the form $y10^{|y|^{2k}}$ for $|y| = \sqrt{n}$. Thus such a Bob gives a (probabilistic) Cook reduction from L instances of length n to at most n^k instances of L of length $O(\sqrt{n})$. We can now use the same Bob recursively to solve the smaller problems. Analyzing the resulting recurrence (and using the fact that the probabilities of error are negligible), we get a net probabilistic running time of $O(n^{2k})$ to decide L, where L was PSPACE complete. ■

Next we show that Bob's running time, as a function of the length of the shortest asymptotically optimal protocol for receiving help from Alice, really needs to be exponential, assuming BPP is different from PSPACE.

Proof (of Theorem 7): Let any $A \in \mathcal{A}|_t$ be given. We start by constructing a family of L-helpful Alices $\{A_\sigma\}_\sigma$ for every $\sigma \in \{0,1\}^*$, where A_σ behaves as follows: if her input is not of the form $\sigma \circ x$, she responds with the empty string. Otherwise, she constructs the following "edited history," and responds as A would have responded given this message history: she removes all pairs of consecutive messages in which Bob's message was not of the form $\sigma \circ x$, and for all remaining messages from Bob, she replaces $\sigma \circ x$ with x.

Clearly, for the Bob B^* that is helped by A and runs in time $O(t(|x|))$, the Bob B^*_σ who converts each query to $\sigma \circ x$ and otherwise computes the same function as B^* has description length $|\sigma| + O(1)$ and runs in time $O(t(|x|))$. Furthermore, every A_σ is in the semantic closure of A, and thus is clearly also in $\mathcal{A}|_t$.

Now suppose there is a L-universal Bob B whose probabilistic running time when interacting with any A_σ on input x is $2^{o(|\sigma|)}|x|^k$, i.e., $O(|x|^k)$ and subexponential in $|\sigma|$. We show that B can be used to decide L, which is assumed to be PSPACE-complete.

We simulate B interacting with an Alice who always responds with the empty string. Notice that if B runs for less than $2^{2k \lg |x|}$ steps, then there is some σ of length $2k \lg |x|$ such that A_σ is consistent with this set of responses. Since B has running time $O(|x|^{1.5k})$ for all such A_σ, for all sufficiently large x B must halt without receiving a response from Alice other than the empty string, and in this case we can output Bob's verdict. Since all A_σ help Bob decide L, this is sufficient. ■

Finally we prove Theorem 6 which shows that even under the "weak" definition of L-universality, Bob is essentially restricted to accepting languages in PSPACE.

Proof (of Theorem 6): We prove this theorem in two steps. First we show that if a Bob B is strongly L'-universal (meeting both the completeness and soundness conditions), then L' must be in PSPACE. We then show that if for some language $L \notin$ PSPACE, Bob meets the completeness condition for L-universality, but is unsound against some Alice, then he is unsound against some helpful Alice.

Step 1: Let B be strongly universal and let A^* be any L'-helpful Alice; since B satisfies the completeness condition, there is some polynomial p such that the interaction $(A^*, B(x))$ halts within $p(|x|)$ steps in expectation for every x.

Now, given x, consider A_x, the Alice who sends B a message that maximizes the probability that B halts within $6p(|x|)$ steps. Since Markov's inequality says that when B interacts with A^* the probability that B takes more than $6p(|x|)$ steps is less than $1/6$, the interaction $(A_x, B(x))$ also halts within $6p(|x|)$ steps with probability greater than $5/6$ (since A_x maximizes this).

Since B satisfies the soundness condition, we find that $(A_x, B(x)) = L'(x)$ within $6p(|x|)$ steps with probability greater than $1/2$. We now observe that in polynomial space, on input x, we can calculate $\Pr[(A_x, B(x)) = 1$ within $6p(|x|)$ steps$]$ where we will decide L' if we accept precisely when this probability is greater than $1/2$.

Step 2: We now consider a language $L \notin$ PSPACE and a Bob B that satisfies the completeness condition of L-universality. Since B can not be sound (by Step 1), we have that there exists A and x such that B halts on interacting with A and $(A, B(x)) \neq L(x)$. We now convert such an Alice A into a helpful one A'_x: Let $\tilde{A}$ be any L-helpful Alice in $\mathcal{A}$ with $\tilde{B}$ being a Bob that can decide L with the help of $\tilde{A}$. Let ℓ be the length of the longest question asked by $B(x)$ when interacting with A. Then A'_x answers queries of length at most ℓ as A does, and answers queries of the form $0^{\ell+1} \circ y$ as $\tilde{A}$ answers y. Clearly A'_x is helpful (to the Bob B'_x who simulates $\tilde{B}$ by padding each query w to $0^{\ell+1} \circ w$). Yet B returns the wrong answer on x when interacting with A'_x violating the completeness condition. Finally note that A'_x is in the semantic closure of $\tilde{A}$. ■

3 Conclusions and Future work

In the previous sections we studied the question, "how can two intelligent interacting players attempt to achieve some meaningful communication in a universal setting, i.e., one in which the two players do not start with a common background?" We return now to the motivation for studying this question, and the challenges that need to be dealt with to address the motivations.

3.1 Practical motivation

We believe that this work has raised and addressed some fundamental questions of intrinsic interest. However this is not the sole motivation for studying this problem. We believe that these questions

also go to the heart of "protocol issues" in modern computer networks. Modern computational infrastructures are built around the concept of communication and indeed a vast amount of effort is poured into the task of ensuring that the computers work properly as communication devices. Yet as computers and networks continue to evolve at this rapid pace, one problem is becoming increasingly burdensome: that of ensuring that every pair of computers is able to "understand" each other, so as to communicate meaningfully.

Current infrastrusctures ensure this ability for pairs to talk to each other by explicitly going through a "setup" phase, where a third party who knows the specifications of both elements of a pair sets up a common language/protocol for the two to talk to each other, and then either or both players learn (download) this common language to establish communication. An everyday example of such an occurence is when we attempt to get our computer to print on a new printer. We download a device driver on our computer which is a common language written by someone who knows both our computer and the printer.

We remark that this issue is a fundamental one, and not merely an issue of improper design. Current protocols are designed with a fixed pair of types of computers in mind. However, we expect for our computers to be capable of communicating with all other communication devices, even ones that did not exist when our computer was built. While it would be convenient if all computers interacted with each other using a single fixed protocol that is static over time, this is no more reasonable to expect than asking humans to agree on a single language to converse in, and then to expect this language to stay fixed over time. Thus, to satisfy our expectations in the current setting, it is essential that computers are constantly updated so as to have universal connectivity over time.

The current model for maintaining connectivity is based (implicitly) on trusted "third parties" who advise us on when to update our languages. This process, however, leads to compromises in terms of *efficiency* as computers spend more of their time downloading languages and less on real computation or communication; *reliability*, since the protocols often lead to inadvertent errors; and *security*, because many viruses and other corrupting elements are introduced at this stage. In particular defining interfaces/protocols is an extremely complex task, made especially so because the players at the two ends are "universal computers" and could behave in any of a countable number of different ways. Anticipating all possible behaviors and reasoning about them is a daunting task, to say the least.

This work was motivated by a somewhat radical alternative scenario for communication. Perhaps we should not set computers up with common languages, but rather exploit the universality in our favor, by letting them evolve to a common language. But then this raises issues such as: how can the computers know when they have converged to a common understanding? Or, how does one of the computers realize that the computer it is communicating with is no longer in the same mode as they were previously, and so the protocol for communication needs to be adjusted? The problem described in the opening paragraph of the introduction is simply the extremal version of such issues, where the communicating players are modeled as having no common background.

Of course, we did not settle such questions in this first work! The aim of this work is to provide a precise formulation of such questions. Perhaps the main contribution of this work is to suggest that communication is not an end in itself, but rather a means to achieving some general goal. Such a goal certainly exists in all the practical settings above, though it is no longer that of deciding membership in some set S. Our thesis is that one can broaden the applicability of this work to other settings by (1) precisely articulating the goal of communication in each setting and (2) constructing "universal protocols" that achieve these goals. In upcoming work we consider some generalizations, which we describe next.

3.2 Forthcoming parts of this work

In upcoming work(s) we consider two extensions of this work.

In the first we consider a broader class of "informational goals" that Bob may wish to achieve by interacting with Alice. For instance, we consider a setting in which Alice is also computationally bounded and can only solve problems in P. In such a setting Bob can no longer expect to gain computational "wisdom" from Alice since he can solve all the problems she can. Yet, a vast fraction of human communication occurs in this setting, and it seems unreasonable to declare all of it a waste. Presumably such interactions satisfy some goal other than that of seeking "wisdom." In upcoming work, we attempt to describe some goals that attempt to model such "intellectual curiosity" and show how to achieve universal communication in such settings.

In the second extension, we consider a generic abstraction of the notion of "goals of communnication." This setting attempts to present a single formulation of a "generic goal of communication" which includes both intellectual goals, such as the one considered in this work, as well as more control-oriented goals, such as that of a computer printing on a printer. We show that whenever such generic goals are verifiable by "Bob," universal communication is possible.

3.3 Open Problems

Given that the task of correctly printing on a printer is a "verifiable" task (at least with some human intervention), the second extension described above should have completely settled the task of designing a universal printing protocol. Yet obviously this is not so. Why? The principal bottleneck to converting any of the proposed approaches of this work and those in the forthcoming ones is that they all employ brute force enumeration of all possible protocols as a means for searching for the right one. Can this be avoided or otherwise ameliorated? The lower bound from Theorem 7 says that this exponential lower bound on the running time can not be avoided. But it does so under our definition of "semantic closure" of Alices. Clearly this lower bound is a consequence of this somewhat broad view of reasonable "Alices." Our lack of a narrow yet functionally rich class of Alices that would allow *efficient* universal communication is, in our opinion, a major challenge to enabling universal communication. At the moment we do not know if any such class can be characterized by reasonable definitions.

Even if such a definition does not exist, the utilization of universal protocols in practical settings is not ruled out entirely. One of the implicit suggestions in this work is that in practical settings, communicating players should periodically test to see if the assumption of common understanding still holds. When this assumption fails, presumably this happened due to a "mild" change in the behavior of one of the players. It may be possible to design communication protocols that use such a "mildness" assumption to search and re-synchronize the communicating players where the "exponential search" takes time exponential in the amount of change in the behavior of the players. Again, pinning down a precise measure of the change and designing protocols that function well against this measure are open issues.

Acknowledgments

We would like to thank Silvio Micali and Ronitt Rubinfeld, for encouraging us through early stages of this work. We are deeply indebted to Oded Goldreich for his enthusiastic support of this work and for the wisdom he shared with us. His detailed comments and suggestions clarified our work to us, and hopefully have helped improved the writeup. We'd like to thank Bob Berwick for his interest

and his efforts to educate us on the work in linguistics, and many illuminating conversations. We would also like to thank Manuel Blum and Steven Rudich for illuminating conversations. Finally, we would like to thank Ryan Williams for taking the time to read and provide comments on an earlier draft of this writeup.

References

[1] Manuel Blum and Sampath Kannan. Designing programs that check their work. In *Proc. 21st STOC*, pages 86–97, 1989.

[2] Hans Freudenthal. *LINCOS: Design of a Language for Cosmic Intercourse*. North-Holland Publishing Company, Amsterdam, 1960.

[3] Oded Goldreich. Personal communication, February 2007.

[4] Oded Goldreich and Dana Ron. On universal learning algorithms. *Information Processing Letters*, 63:131–136, 1997.

[5] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[6] Leonid A. Levin. Universal search problems. *Probl. Inform. Transm.*, 9:265–266, 1973.

[7] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, October 1992.

[8] Willard Van Orman Quine. *Word and Object*. MIT Press, Cambridge, 1960.

[9] Adi Shamir. IP = PSPACE. *JACM*, 39(4):869–877, 1992.

[10] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[11] Andrew C.-C. Yao. Some complexity questions related to distributed computing. In *Proc. 11th STOC*, pages 209–213, 1979.

A On LINCOS

The aim of LINCOS [2] is to provide a means to send meaningful messages across intergalactic distances to "humanlike" recipients with whom we have had no prior contact. Because our current understanding of the laws of physics suggests that communication across such vast distances is necessarily slow, it is essentially necessary that the method be non-interactive (in contrast to the protocols we have considered). Thus, over the course of a transmission, Freudenthal attempts to develop a vocabulary, beginning with simple, concrete concepts, and gradually building on these to introduce more and more abstract concepts, until a sufficiently rich vocabulary has been developed to permit the desired message to be sent in a form that the recipient will understand.

Freudenthal assumes that the message will be transmitted by means of radio waves, and uses physical properties of the waves to illustrate the earliest concepts (again, in contrast to our setting of pure information, where we have abstracted away these properties). For example, a pulse of a given duration is used to illustrate a length of time, and a natural number n is initially illustrated

by n pulses. New vocabulary are introduced by including a new "symbol" (waveform) in a list of example messages illustrating the use of that symbol. Again, for example, suppose that we wish to define a symbol that means "equals." Let $\cdot$ denote a pulse, and let $=$ denote the waveform we wish to associate with the notion of equality. We would then send a message of the following form, letting spaces be communicated by short pauses and new lines be communicated by longer pauses:

$$\cdot = \cdot$$
$$=$$
$$\cdot\cdot\cdot\cdot\cdot = \cdot\cdot\cdot\cdot\cdot$$
$$\cdot\cdot\cdot\cdot\cdot\cdot\cdot = \cdot\cdot\cdot\cdot\cdot\cdot\cdot$$
$$\cdot\cdot\cdot = \cdot\cdot\cdot$$
$$\vdots$$

After many, many such examples, roughly corresponding to messages of the form "$n = n$" for various values of n, Freudenthal asserts that a "humanlike" recipient will have observed that the quantities preceding and following the waveform for $=$ are always equal, and has inferred that "=" is generally used when these quantities are equal. Therefore, we assume that henceforth the symbol "=" may be used to communicate the concept of equality.

Our primary criticism of LINCOS is that Freudenthal does not define precisely what his setting is, where in the absence of formal definitions, there is no way to prove that the intentions are achieved. If we attempt to provide a formal setting for LINCOS, we quickly see that there are some unresolved issues, as follows. Aside from taking basic, concrete notions (pauses, natural numbers, etc.) for granted, we may observe that the central assumptions of LINCOS are that:

1. Sufficiently many examples are given so that the intended concept is "obvious"
2. The concept being illustrated is associated with the symbol being introduced

Where, even if we take the subtle issue in assumption 2 for granted, it is still unclear how to determine in general the number of examples necessary for assumption 1 to hold. A more rigorous analysis of LINCOS would be desirable, but this seems to be beyond our capabilities at the moment.