Interpreting Sequent Calculi as Client-Server Games

Christian G. Fermüller and Timo Lang

TU Vienna, Austria

Abstract. Motivated by the interpretation of substructural logics as resource-conscious reasoning, we introduce a client-server game characterizing provability in single-conclusion sequent calculi. The set up is modular and allows to capture multiple logics, including intuitionistic and (affine) linear intuitionisitic logic. We also provide a straightforward interpretation of subexponentials, and moreover introduce a game where the information provided by the server is organized as a stack, rather than as a multiset or list.

1 Introduction

Resource consciousness is routinely cited as a motivation for considering substructural logics (see, e.g., [10]). But usually the reference to resources is kept informal, like in Girard's well-known example of being able to buy a pack of Camels and/or a pack of Marlboro [5] with a single dollar, illustrating linear implication as well as the ambiguity of conjunction between the "multiplicative" and "additive" reading. The invitation to distinguish, e.g., between a "causal", action-oriented interpretation of implication and a more traditional understanding of implication as a timeless, abstract relation between propositions is certainly inspiring and motivating. However, the specific shape and properties of proof systems for usual substructural logics owe more to a deep analysis of Gentzen's sequent system than to action-oriented models of handling scarce resources of a specific kind. Various semantics, in particular so-called game semantics for (fragments of) linear logics [1, 3] offer additional leverage points for a logical analysis of resource consciousness. But these semantics hardly support a straightforward reading of sequent derivations as actions plans devised by resource conscious agents. Moreover, the inherent level of abstraction often does not match the appeal of (e.g.) Girard's very concrete and simple picture of action-oriented inference.

We introduce a two-player game based on the idea that a proof is an *action-plan*, i.e. a *strategy* for one of the players (the "Client") to reduce particular *structured information* to information provided by the other player (the "Server"). As we will show, the interpretation of game states as single conclusion sequents leads to variations of the basic game, that match (affine) intuitionistic linear logic, but also other substructural logics. To emphasize the indicated shift of perspective, relative to traditional interpretations of formulas as sentences or propositions or

types, we introduce the notion of an *information package*, which emphasizes the interpretation of formulas as (in general) compound information, that is built up from atomic pieces of information using constructors that indicate possible ways of accessing the information.

Obviously our Client-Server games constitute a variant of *game semantics*; therefore a few words on the relation to other forms of game semantics are appropriate. Already in the late 1950s Lorenzen [9] proposed to *justify* intuitionistic logic in terms of dialogue game, where a proponent defends a statement against systematic attacks by an opponent. Logical validity is identified with the existence of a winning strategy for the proponent. This setup has later been generalized to other logics; see, e.g., [7, 11]. While there are some obvious similarities between Lorenzen-style dialogue games and our Client-Server games the different motivation triggers also structural differences. In particular, Lorenzen and his followers argue that the two players should have 'equal rights': not only the specific rules for the logical connectives, but also the so-called frame rules, that regulate the overall progression of a dialogue, should be as symmetric as possible. In contrast, we deliberately break this symmetry and view the Client as the active 'scheduler' of the interaction with a largely passive or at least dis-interested Server. Similar remarks hold for game semantics developed for (fragments of) linear logic in the wake of [1,3,8]. The idea there is to view propositions as games and connectives as operators on games. Again, the symmetry between the two players is important, as witnessed by the prominence of the copy-cat strategy, which has no counterpart in our Client-Server games. Finally, Japaridze's Computability Logic [6] deserves to be mentioned, where formulas are interpreted as computational problems. The underlying model of interactive computation is a game between a machine and the environment. While somewhat related in spirit to our (much simpler and more specific) game model, the corresponding logics and inference mechanisms are again quite different. Probably the most important feature of our approach is that we aim at a *direct interpretation* of sequent rules as rules for systematically reducing information packages to its components.

The paper is structured as follows: In Section 2, we introduce our client-server game in its basic form. In Section 2, we show that this game captures provability in intuitionistic logic. Section 4 describes a resource-aware version of the game, which is shown to capture affine intuitionistic logic. In Section 5, we make some remarks on the interpretation of (sub)exponentials. The final section 6 discusses a variant of the game where information packages are arranged in a stack.

2 A Client-Server Game for Intuitionistic Logic

In our $\mathbf{C}/\mathbf{S}$(I)-game, a *client* $\mathbf{C}$ maintains that the information packaged as G can be obtained from the information represented by the packages $F_1, \ldots, F_n$, provided by a *server* $\mathbf{S}$, via stepwise reduction of complex information packages (henceforth short *ips*, singular *ip*) into simpler ones. At any state of the game, the *bunch of information provided by* $\boldsymbol{S}$ is a (possibly empty) multiset of ips.

The ip G which **C** currently claims to be obtainable from that information is called **C**'s *current ip*. The corresponding state is denoted by

$$F_1, \ldots, F_n \rhd G.$$

The game proceeds in rounds that are always initiated by **C** and, in general, solicit some action from **S**. We look at the game from the client's point of view.[1] There are two different types of requests that **C** may submit to S: (1) UNPACK an ip provided by the server, and (2) CHECK my (i.e. the clients) current ip. We call the ip chosen by **C** for either the UNPACK- or CHECK-request the *active ip*. Thus in a CHECK-request the active ip is always **C**'s current ip. Both UNPACK- and CHECK-requests depend on the *structure* of the active ip. For now, we will consider the following types of ips:

- *atomic* ips, which admit no further reduction
- among those, a special ip $\bot$, denoting an elementary *inconsistency*
- *complex* ips which are build from simpler ips by means of the constructors $\wedge$, $\vee$, and $\to$ (called *any of*, *some of* and *given* respectively).

We use lowercase letters a, b, c for for atomic ips and uppercase letters F, G, H, K for ips which may be either complex or atomic. Multisets of ips are denoted by Γ or Δ. The rules for reducing complex ips are given in Table 1. One may easily introduce other constructors for complex ips into the game by specifying their UNPACK- and CHECK-rules, and we will see some examples of that later.

At the beginning of each round of the game **C** is free to choose whether she wants to continue with a request of type UNPACK (if possible) or of type CHECK; moreover in the first case **C** can freely choose any occurrence of a non-atomic ip or an occurrence of $\bot$ in the bunch of information provided by **S**. Formally, each initial state $F_1, \ldots, F_n \rhd G$ induces an extensive two-players win/loose (zero sum) game of perfect information in the usual game theoretic sense.

The corresponding game tree is finitely branching, but may be infinite since C may request to unpack the same ip repeatedly. We will look at strategies only at the level of states resulting from fully completed rounds. A winning strategy τ for **C** can therefore be identified with a finite, downward growing, rooted tree of game states, where all leaves are winning states for **C** according to either rule (CHECK a) or (UNPACK $\bot$). The root of τ is the initial state of the relevant instance of the **C**/**S**(I)-game in question. When, at a state S, the strategy τ tells **C** to continue with a round of type (UNPACK $F_1 \to F_2$) or (CHECK $F_1 \wedge F_2$), then τ branches at S into two successor states according to the possible choices available to **S** as specified by the rules. On the other hand, no branching occurs at states where τ tells **C** to continue according to any other rule, since those rules do not involve a choice of **S**.

The game rules are *local*: the validity of a move of **C** only depends on the presence of a certain ip in the current game state, but not on the complete bunch

[1] Since we only care about winning strategies for **C**, the server **S** may be viewed as acting nondeterministically or probabilistically, if preferred.

a	atomic ip
	UNPACK: - not possible -
	CHECK: The game ends and **C** wins iff a is contained in the bunch of information proved by **S**.
$\bot$	inconsistency
	UNPACK: The game ends and **C** wins.
	CHECK: as in the case for atomic F
$F_1 \wedge F_2$	any of F_1, F_2
	UNPACK: **C** chooses an ip out of $\{F_1, F_2\}$ which **S** then has to add to the bunch of provided information
	CHECK: **S** chooses an ip out of $\{F_1, F_2\}$ and sets it as **C**'s new current ip.
$F_1 \vee F_2$	some of F_1, F_2
	UNPACK: **S** chooses an ip out of $\{F_1, F_2\}$ and adds it to the bunch of provided information
	CHECK: **C** chooses an ip out of $\{F_1, F_2\}$ and sets it as the new current ip.
$(F_1 \to F_2)$	F_2 given F_1
	UNPACK: **S** chooses whether to add F_2 to the bunch of provided information, or to force **C** to replace its current ip by F_1.
	CHECK: F_1 is added to the bunch of provided information and **C**'s current ip is replaced by F_2.

Table 1: Atoms, constructors and rules for $\mathbf{C}/\mathbf{S}(\mathrm{I})$

of provided information. Furthermore, **S**'s moves are restricted to ips previously chosen by **C**. From these observations, it follows easily that:

Proposition 1 *If* $\boldsymbol{C}$ *has a winning strategy in the* $\boldsymbol{C}/\boldsymbol{S}(I)$*-game* $\Gamma \rhd F$ *and* Δ *is any multiset of ips, then* $\boldsymbol{C}$ *also has a winning strategy in* $\Delta, \Gamma \rhd F$*.*

Proof. Let τ be any winning strategy for **C** in $\Gamma \rhd F$. By the locality of the rules, **C** can also move according to τ in the extended game $\Delta, \Gamma \rhd F$, and if she does so, **S** will not have more choices than before. Hence τ is still a *strategy* in the extended game. Since ips in Δ are never unpacked while moving according to τ, and since τ is a winning strategy in $\Gamma \rhd F$, the strategy leads in the extended game to states of the form $\Delta, \Gamma' \rhd F'$ where $\Gamma' \rhd F'$ is a winning state for **C**. Since also the winning conditions of the $\mathbf{C}/\mathbf{S}(\mathrm{I})$-game are local, $\Delta, \Gamma' \rhd F'$ is still a winning state for **C**. □

3 The adequateness of C/S(I) for Intuitionistic Logic

Let us now identify atomic ips with propositional variables and complex ips with their corresponding propositional formulas. It is well-known that we may read winning strategies for **C** as proofs in a sequent calculus, where the turnstile $\Rightarrow$ stands for $\rhd$ and the inital sequents correspond to winning states. In our case, the initial sequents are thus

$$\frac{}{\Gamma, a \Rightarrow a} \qquad \text{and} \qquad \frac{}{\Gamma, \bot \Rightarrow F}$$

corresponding to the states $\Gamma, a \rhd a$ (where **C** wins by sending a (CHECK a)-request) and $\Gamma, \bot \rhd F$ (where **C** wins by sending an (UNPACK $\bot$)-request). The UNPACK-rule for $\vee$ translates to the sequent rule

$$\frac{\Gamma, F_1 \vee F_2, F_1 \Rightarrow H \qquad \Gamma, F_1 \vee F_2, F_2 \Rightarrow H}{\Gamma, F_1 \vee F_2 \Rightarrow H}$$

where the two premises correspond to the two possible choices of **S**. The CHECK-rule for $\vee$ translates to the pair of rules

$$\frac{\Gamma \Rightarrow F_1}{\Gamma \Rightarrow F_1 \vee F_2} \qquad \text{and} \qquad \frac{\Gamma \Rightarrow F_2}{\Gamma \Rightarrow F_1 \vee F_2}$$

corresponding to the two possible choices of **C**. Similarly, one writes down the sequent rules for the remaining connectives $\wedge, \rightarrow$. Using this translation, the rules and initial sequents exactly match the sequent calculus **LIk** for intuitionistic logic (cf. [12]). We thus obtain:

Theorem 2 *The following are equivalent:*

1. ***C** has a winning strategy in the **C/S**(I)-game $\Gamma \rhd H$*
2. **LIk** $\vdash \Gamma \Rightarrow H$
3. $(\bigwedge \Gamma \Rightarrow H)$ *is intuitionistically valid* [2].

LIk arises from the traditional sequent calculus **LI** for intuitionistic logic by eliminating contraction by building into the logical rules and eliminating weakening by generalizing the initial sequents (axioms) correspondingly [3].

We get a game directly matching the rules for **LI** by making the following modifications to the **C**/**S**(I)-game: First, we change the UNPACK-rules such that the active ip is *removed* from the bunch of provided information after use; second, we add two types of request called DISMISS and COPY, which allow **C** to either remove or duplicate ips from the bunch of provided information: and finally we allow only

$$a \rhd a \qquad \text{and} \qquad \bot \rhd F$$

[2] $\bigwedge \Gamma$ denotes the conjunction of all formulas in Γ.

[3] We assume that **LI** is already formulated using multisets - otherwise, this would be another difference between the calculi.

as winning states for **C**. Let us call the modified game $\mathbf{C}/\mathbf{S}(\mathrm{I})^*$.

Via Theorem 2, results from the structural proof theory of **LIk** or **LI** turn into statements about winning strategies in $\mathbf{C}/\mathbf{S}(\mathrm{I})$ or $\mathbf{C}/\mathbf{S}(\mathrm{I})^*$. As a simple example (which works for either variant of the calculus/game), the *soundness* of the rule

$$\frac{\Gamma \Rightarrow F \qquad \Gamma \Rightarrow G}{\Gamma \Rightarrow F \wedge G}\ (\wedge\mathrm{R})$$

says that if **C** has a winning strategy τ for $\Gamma \rhd F$ and σ for $\Gamma \rhd G$, then she has a winning strategy in $\Gamma \rhd F \wedge G$. The winning strategy, of course, is this: In her first move, **C** sends a (CHECK $F \wedge G$) request. If now **S** chooses F, the game is a $\Gamma \rhd F$ where she can play according to τ to win; otherwise, if **S** picks G, she plays according to σ.

More interestingly, the *invertibility* of the ($\wedge$R) rule – the fact that the validity of its conclusion implies the validity of its premises – says that if **C** has a winning strategy in $\Gamma \rhd F \wedge G$, then she has such a winning strategy where her first move is (CHECK $F \wedge G$).

The correspondence of Theorem 2 goes both ways: For example, Proposition 1 is nothing but a game theoretic proof of the admissibility of the weakening rule in **LIk**. As another toy application, we give a proof of cut-admissibility for the $\rightarrow$-free fragment of **LI** by using the game semantics of $\mathbf{C}/\mathbf{S}(\mathrm{I})^*$ (of course, cut is admissible also in the full language).

Proposition 3 *Assume that $\rightarrow$ does not appear in Γ, Δ, G, H. If **C** has winning strategies in the $\mathbf{C}/\mathbf{S}(\mathrm{I})^*$-games $\Gamma \rhd G$ and $G, \Delta \rhd H$ then she also has a winning strategy in $\Gamma, \Delta \rhd H$.*

Proof. Let τ be a winning strategy for $\Gamma \rhd G$ and σ a winning strategy for $G, \Delta \rhd H$. We proof by induction on the structure of G that **C** wins in $\Gamma, \Delta \rhd H$.

1. $G \equiv a$ for atomic a: Since the game ends when atomic ips are checked, all but the last move in τ must be UNPACK-requests. Since τ is winning, a play on $\Gamma \rhd a$ according to τ always ends in a state of the form $\bot \rhd a$ or $a \rhd a$. **C** can thus move according to τ in the game $\Gamma, \Delta \rhd H$ to arrive at a state $\bot, \Delta \rhd H$ or $a, \Delta \rhd H$. In the first case she wins by sending DISMISS-requests repeatedly until she is in the winning state $\bot \rhd H$. In the second case, she can move according to σ to win.
2. $G \equiv F_1 \wedge F_2$: **C** starts moving according to τ in the game $\Gamma, \Delta \rhd H$ until a (CHECK $F_1 \wedge F_2$)-request appears (if that does not happen, the game must arrive eventually at a state $\Gamma', \bot, \Delta \rhd H$ where **C** wins). The game is now in a state $\Gamma', \Delta \rhd H$. Note that **C** must have winning strategies in $\Gamma' \rhd F_1$ and $\Gamma' \rhd F_2$, since by moving according to τ in the game $\Gamma \rhd F_1 \wedge F_2$ she ends up in a state $\Gamma' \rhd F_1 \wedge F_2$ and now, since the next step in τ is (CHECK $F_1 \wedge F_2$), **C** must be prepared for any choice of F_1, F_2 by **S**.
   Back to the game state $\Gamma', \Delta \rhd H$. Here, **C** now switches to the strategy σ and moves until an (UNPACK $F_1 \wedge F_2$)-request appears (again, if this does not

happen, the game must have arrived in a winning state). The game is then in a state $\Gamma', \Delta' \rhd H'$. Without loss of generality, let us assume that σ tells **C** to pick F_1 in the rule for $\wedge$. Then **C** has a winning strategy $\Delta', F_1 \rhd H'$, because this state arises by starting in $F_1 \wedge F_2, \Delta \rhd H$ and moving according to the winning strategy σ.
Applying the induction hypothesis to the states $\Gamma' \rhd F_1$ and $\Delta', F_1 \rhd H'$ (and their respective winning strategies), we thus know that **C** has a winning strategy in $\Gamma', \Delta' \rhd H'$, and so she can continue playing.

3. $G = F_1 \vee F_2$: similar to the previous case. □

Remark 4 *Note that the length of play according to the winning strategy constructed in the above proof is polynomially bounded in the length of plays according to the winning strategies for $\Gamma \rhd G$ and $G, \Delta \rhd H$. This cannot be the case if we include $\rightarrow$, since it is known that cut reduction in the full fragment of intuitionistic logic increases proof size exponentially.*

4 Resource Consciousness

Probably the most important step in turning the **C**/**S**(I)-game into a 'resource conscious' one, regards rules that entail a choice by **S** and thus require **C** to be prepared to act in more than just one possible successor state to the current state. The **C**/**S**(I)-rules allow **C** to use all the information provided by **S** in each of the possible successor states. If, instead, we require **C** to declare which ips she intends to use for which of those options – taking care that she is using each occurrence of an ip exactly once – then we arrive at rules that match multiplicative instead of additive connectives.

Following the tradition of linear logic, we do not discard the previously defined rules, but rather extend the game by new ip constructors and their corresponding resource concious rules. We also introduce a unary 'safety' constructor ! (called *exponential* in the literature on linear logic). Ips prefixed by ! are meant to be exempt from resource consciousness and thus behave like ips in the **C**/**S**(I)-game. Ips not prefixed by ! are called *unsecured*. Table 2 lists all new constructors and their corresponding rules[4]. Let us denote by **C**/**S**(IAL) the following modification of game **C**/**S**(I):

1. Constructors and rules for $\otimes, \multimap$ and ! are added as in Table 2
2. The UNPACK-rules for $\wedge, \vee$ and $\rightarrow$ are changed so that the active formula is *removed* at the end of the request.

We claim that the logic captured by **C**/**S**(IAL) is *intuitionistic affine logic* **IAL**, i.e. intuitionistic linear logic with weakening [5]. A standard sequent calculus for **IAL** is presented in Table 3. We need the following preliminary result analogous to Proposition 1:

[4] In these rules, operations as *replacing* and *removing* an ip in a multiset are meant to affect only the active *instance* of the ip, rather than all instances of the ip in the multiset.

$\otimes(F_1, \ldots, F_n)$	each of $F_1, \ldots, F_n$
	UNPACK: $\otimes(F_1, \ldots, F_n)$ is replaced by $F_1, \ldots, F_n$
	CHECK: **C** marks every unsecured ip in the bunch of provided information with one of $F_1, \ldots F_n$. Next, **S** chooses one F_i out of $F_1, \ldots F_n$. Then **C**'s current ip is changed to F_i and all unsecured ips not marked with F_i are removed.
$(F_1 \multimap F_2)$	F_2 from F_1
	UNPACK: **C** marks every unsecured ip in the bunch of provided information (except the instance of $(F_1 \multimap F_2)$) with either F_1 or F_2. **S** then chooses between the premise F_1 and the conclusion F_2. If **S**'s choice was F_1, **C**'s current ip is changed to F_1 and all ips marked with F_2 are removed. If **S**'s choice was F_2, F_2 is added to the bunch of provided ips and all ips marked with F_1 are removed. In any case, the instance of $F_1 \multimap F_2$ is removed as well.
	CHECK: F_1 is added to the bunch of provided information and **C**'s current ip is replaced by F_2.
$!F$	safe F
	UNPACK: A copy of F is added to the bunch of provided information, and then an UNPACK-request is performed on this copy.
	CHECK: All unsecured ips are removed, and **C**'s current ip is changed to F.

Table 2: Resource-conscious rules in **C**/**S**(IAL)

Proposition 5 *If **C** has a winning strategy in the **C**/**S**(IAL)-game $\Gamma \rhd F$ and Δ is any multiset of ips, then **C** also has a winning strategy in $\Delta, \Gamma \rhd F$.*

Proof. By induction on the number of steps in a winning strategy for $\Gamma \rhd F$. We only consider the case that the first step in the winning strategy is to send a (CHECK $!G$)-request. The state must thus be of a form $!\Gamma_1, \Gamma_2 \rhd !G$, where we assume that all ips in Γ_2 are unsecured ($!\Gamma$ denotes $\{!F \mid F \in \Gamma\}$). The request results in the state $!\Gamma_1 \rhd G$, for which **C** therefore has a winning strategy. It follows that **C** wins in $\Delta, !\Gamma_1, \Gamma_2 \rhd !G$: She starts by sending a (CHECK $!G$)-request, resulting in the state $\Delta_1, !\Gamma_1 \rhd G$, where Δ_1 denotes the set of all safe formulas in Δ. Since **C** has a winning strategy for $!\Gamma_1 \rhd G$, the induction hypothesis implies that she also wins in $\Delta_1, !\Gamma_1 \rhd G$. □

Theorem 6 *The following are equivalent:*

1. ***C** has a winning strategy in the **C**/**S**(IAL)-game $\Gamma \rhd H$*
2. **IAL** $\vdash \Gamma \Rightarrow H$

$$\frac{}{a \Rightarrow a}\ (\mathrm{id}) \qquad \frac{}{\bot \Rightarrow A}\ (\bot) \qquad \frac{\Gamma \Rightarrow A}{B, \Gamma \Rightarrow A}\ (\mathrm{W})$$

$$\frac{\Gamma, A_i \Rightarrow C}{\Gamma, A_1 \wedge A_2 \Rightarrow C}\ (\wedge \mathrm{L}i)\ i = 1, 2 \qquad \frac{\Gamma \Rightarrow A \qquad \Gamma \Rightarrow B}{\Gamma \Rightarrow A \wedge B}\ (\wedge \mathrm{R})$$

$$\frac{\Gamma, A \Rightarrow C \qquad \Gamma, B \Rightarrow C}{\Gamma, A \vee B \Rightarrow C}\ (\vee \mathrm{L}) \qquad \frac{\Gamma \Rightarrow A_i}{\Gamma \Rightarrow A_1 \vee A_2}\ (\vee \mathrm{R}i)\ i = 1, 2$$

$$\frac{\Gamma, A, B \Rightarrow C}{\Gamma, A \otimes B \Rightarrow C}\ (\otimes \mathrm{L}) \qquad \frac{\Gamma \Rightarrow A \qquad \Delta \Rightarrow B}{\Gamma, \Delta \Rightarrow A \otimes B}\ (\otimes \mathrm{R})$$

$$\frac{\Gamma \Rightarrow A \qquad \Delta, B \Rightarrow C}{\Gamma, \Delta, A \multimap B \Rightarrow C}\ (\multimap \mathrm{L}) \qquad \frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow A \multimap B}\ (\multimap \mathrm{R})$$

$$\frac{\Gamma, !A, !A \Rightarrow B}{\Gamma, !A \Rightarrow B}\ (!\mathrm{C}) \qquad \frac{\Gamma, A \Rightarrow B}{\Gamma, !A \Rightarrow B}\ (!\mathrm{dR}) \qquad \frac{!\Gamma \Rightarrow A}{!\Gamma \Rightarrow !A}\ (!\mathrm{R})$$

Table 3: The sequent calculus **IAL**

Proof. (Sketch) Again, we use the correspondence between winning strategies and proofs described in Section 3. However, the game rules do not directly match the rules of **IAL** in all cases, thus we have to provide some further arguments.

First, there is no game rule corresponding to weakening (W). This is not a problem, because weakening is admissible in the game theoretic version of the rules by Proposition 5.[5]

Second, there is no game rule corresponding to (!C). Rather, the splitting in multiplicative rules is changed so that safe formulas never need to be split, making the duplication of safe formulas obsolete. The equivalence of the thus obtained calculus is known in the literature (see for example the *dyadic* calculus of [2]).

Finally, the (Unpack $!F$)-rule in our game semantics forces us to immediately unpack the copy of F after it has been created. There is no such requirement in **IAL**: here we may create a copy of a safe formula by a combination of (!C) and (!dR), which might be used only later in a proof (if at all). It is however not hard to check that such a detour is never necessary. This can also be seen as a special case of Andreoli's results on *Focusing* [2]. □

[5] We remark that (W) is *not* admissible in **IAL**, even if one relaxes the axioms, because of the (!R)-rule. Our corresponding (Check $!F$)-rule is different: It could be written as

$$\frac{!\Gamma \Rightarrow F}{\Delta, !\Gamma \Rightarrow !F}$$

which has a built-in weakening.

Before closing this section, let us remark that we also obtain a game adequate for **ILL** (full intuitionistic linear logic) by allowing only

$$a \rhd a \qquad \text{and} \qquad \bot \rhd F$$

as winning states for **C** and introducing atomic ips $0, 1, \top$ with their corresponding rules. This amounts to an interpretation of sequents as **C**/**S**-game states, where **C** announces that she needs *precisely* the information provided by **S** to obtain her current ip.

5 Interpreting exponentials and subexponentials

The UNPACK-rule for ! (together with the CHECK-rule for $\otimes$ and the UNPACK-rule for $\multimap$) shows that safe ips are exempt from resource consciousness: operations are performed on copies of the safe ip rather than on the ip itself. The UNPACK-rule for ! says that the safety predicate is *hereditary*: If F can be demonstrated from a bunch of safe ips, then F is also safe.

C can send (UNPACK $!F$)-requests to the same ip $!F$ as often as she wishes. Furthermore, if **C** has a winning strategy for $\Gamma \rhd !F$ then she also has winning strategies for $\Gamma \rhd F^{\otimes n}$ for any n, where $F^{\otimes n}$ denotes $\underbrace{F \otimes \ldots \otimes F}_{n}$. This is most easily seen by first checking that **C** has a winning strategy in $!F \rhd F^{\otimes n}$ and then using the fact that the cut rule is admissible in **IAL**.

The meaning of $!F$ is often paraphrased as 'arbitrarily many F'. But this intuition is not without pitfalls, as the observation demonstrates.

Lemma 7 *Assume $a, b \neq \bot$. **C** has a winning strategy in $a, !(a \multimap a \otimes b) \rhd b^{\otimes n}$ for any n, but she has no winning strategy in $a, !(a \multimap a \otimes b) \rhd !b$.*

Formulated proof-theoretically, Lemma 7 entails that the infinitary rule

$$\frac{\Gamma \Rightarrow F^{\otimes n} \text{ for all } n}{\Gamma \Rightarrow !F}\ (!\mathrm{R}^{\omega})$$

is not admissible in IAL. The interpretation of ! is improved by thinking of $!F$ not as arbitrarily many F's, but as a *single* container containing (potentially) arbitrarily many F's. The problem is that this does not tell us much about what we should require from a proof of $!F$.

Instead, we invite the reader to think of the rules for the safety predicate as (partially) specifying a concept of safety, where being exempt form consumption through unpacking (i.e., resource-consciousness) is the essential minimal requirement. This also aligns with the observation that when adding another unary constructor $!'$ with the same rules as ! to **IAL**, one cannot[6] prove the equivalence of ! and $!'$. Variants of the standard exponential introduced in this

[6] We remark that the combination of the rules (!C), (!dR) and ($!\mathrm{R}^{\omega}$) *does* define an exponential ! uniquely. However, cut is not admissible in the resulting system.

way are usually called *subexponentials*. In the 'arbitrarily many'-interpretation of the exponential, the existence of subexponentials seems to be mysterious – how can there be two *different* concepts of 'arbitrarily many'?

In the safety interpretation, we may think of different subexponentials !'s as corresponding to different *levels of safety*. In fact, we can add constructors $!_1, !_2, \ldots, !_n$, where greater indices denote greater safety. A natural generalization of the !-rule is then the following:

$!_i F$ safety level i for F

UNPACK: A copy of F is added to the bunch of provided information, and then an UNPACK request on this copy is invoked.

CHECK: All ips of safety level *less than* i (including the unsecured ones) are removed, and **C**'s current ip is changed to F.

One may go further and arrange the safety levels in a *partial order* rather than a linear order, with the obvious modification of the (CHECK)-rule. At some point, one looses cut-admissibility of the logic – we refer the reader to [4, Chapter 5].

6 The Server as Stack

In the games considered so far, **C**'s choice of the active ip at the beginning of each round was completely free. We now consider a variant of the game where the bunch of provided information is a list rather than a multiset, and **C** can only access the last element in the list. In other words, we think of the server as a *stack*.

We consider the constructors and game rules as given in Table 4. Note that in UNPACK-requests, the active ip is now always the topmost element of the stack.

Let us call the resulting game **C**/**S**(**STACK**). Again, we translate game states to sequents (which are now *lists* of ips) and game rules to sequent rules. We write stacks from left to right, so that the rightmost element of a list of ips corresponds to the topmost element of the stack. Let us call the resulting system **LSTACK**. The initial sequents are thus

$$\frac{}{\Gamma, a \Rightarrow a} \qquad \text{and} \qquad \frac{}{\Gamma, \bot \Rightarrow F}\ .$$

Of the rules, we only mention those for $\to$ and $(;)$ explicitly. They are

$$\frac{\Gamma, G \Rightarrow H \qquad \Gamma \Rightarrow F}{\Gamma, F \to G \Rightarrow H}\ (\to\mathrm{L}) \qquad \frac{\Gamma_1, F, \Gamma_2 \Rightarrow G}{\Gamma_1, \Gamma_2 \Rightarrow F \to G}\ (\to\mathrm{R})$$

and

$$\frac{\Gamma, G, F \Rightarrow H}{\Gamma, (F;G) \Rightarrow H}\ (;\mathrm{L}) \qquad \frac{\Gamma_2 \Rightarrow F \qquad \Gamma_1 \Rightarrow G}{\Gamma_1, \Gamma_2 \Rightarrow (F;G)}\ (;\mathrm{R})$$

where Γ_1 and Γ_2 correspond to the lower and the upper part of the stack in the rule (CHECK $(F;G)$) respectively.

Analogously to Theorems 2 and 6, we have

a	atomic ip
	UNPACK: - not possible -
	CHECK: The game ends and **C** wins iff a is the topmost item on the stack.
$\bot$	inconsistency
	UNPACK: The game ends and **C** wins.
	CHECK: as in the case for atomic F
$F_1 \wedge F_2$	any of F_1, F_2
	UNPACK: **C** chooses an ip F_i out of $\{F_1, F_2\}$. **S** then has to replace $F_1 \wedge F_2$ by F_i.
	CHECK: **S** chooses an ip out of $\{F_1, F_2\}$ and sets it as **C**'s new current ip.
$F_1 \vee F_2$	some of F_1, F_2
	UNPACK: **S** replaces $F_1 \vee F_2$ by one ip out of $\{F_1, F_2\}$.
	CHECK: **C** chooses an ip out of $\{F_1, F_2\}$ and sets it as the new current ip.
$(F_1 \to F_2)$	F_2 given F_1
	UNPACK: **S** removes $(F_1 \to F_2)$ and chooses whether to add F_2 on top of the stack, or to force **C** to replace its current ip by F_1.
	CHECK: **C** choses a position in the stack at which **S** has to insert F_1, and changes her current ip to F_2.
$(F_1; F_2)$	F_2 after F_1
	UNPACK: **S** replaces $(F_1; F_2)$ by the two ips F_2, F_1 to the stack (so that F_1 becomes the topmost element).
	CHECK: **C** chooses a splitting of the stack into an upper and a lower part (both parts may be empty). **S** then decides whether to change **C**'s current ip to F_1 and continue the game with the upper part of the stack, or to change **C**'s current ip to F_2 and continue the game with the lower part of the stack.

Table 4: Constructors and rules for **C**/**S**(**STACK**)

Theorem 8 *The following are equivalent:*

1. ***C** has a winning strategy in the **C**/**S**(***STACK***)-game $\Gamma \rhd H$.*
2. **LSTACK** $\vdash \Gamma \Rightarrow H$.

The rules for the connective $(;)$ resemble those of the $\otimes$ of linear logic, only that in the right rule, the premises are split in an ordered way. $(;)$ internalizes the lin-

ear order of the stack. It has the following properties, which are straightforward to check:

Proposition 9

1. *(non-commutativity)* $\boldsymbol{C}$ *has* no *winning strategy in* $(F;G) \rhd (G;F)$.
2. *(associativity 1)* $\boldsymbol{C}$ *has a winning strategy in* $(F;(G;H)) \rhd ((F;G);H)$.
3. *(associativity 2)* $\boldsymbol{C}$ *has a winning strategy in* $((F;G);H) \rhd (F;(G;H))$.

Proposition 10

1. $\boldsymbol{C}$ *has a winning strategy in* $\Gamma, F \rhd F$.
2. $\boldsymbol{C}$ *has a winning strategy in* $\Gamma, F, F \to G \rhd G$.

Proof. The proof of (1) proceeds by induction on F. If F is atomic, $\Gamma, F \rhd F$ is already a winning state for **C**. If $F \equiv G \to H$, the **LSTACK**-derivation

$$\dfrac{\dfrac{\Gamma, G, H \Rightarrow H \qquad \Gamma, G \Rightarrow G}{\Gamma, G, G \to H \Rightarrow H}\,(\to\mathrm{L})}{\Gamma, G \to H \Rightarrow G \to H}\,(\to\mathrm{R})$$

demonstrates that **C** can always move to end up in a state $\Gamma, G, H \rhd H$ or $\Gamma, G \rhd G$, for both of which she has winning strategies by the induction hypothesis. If $F \equiv (G;H)$, the **LSTACK**-derivation

$$\dfrac{\dfrac{G \Rightarrow G \qquad \Gamma, H \Rightarrow H}{\Gamma, H, G \Rightarrow (G;H)}\,(;\mathrm{R})}{\Gamma, (G;H) \Rightarrow (G;H)}\,(;\mathrm{L})$$

demonstrates that **C** can always move to end up in a state $G \rhd G$ or $\Gamma, H \rhd C$, and again she has winning strategies for both states by the induction hypothesis. The other cases are similar.

For (2), **C** starts the game $\Gamma, F, F \to G \rhd G$ by sending an (Unpack $F \to G$)-request. Depending on the subsequent choice of **S**, the game is then either in the state $\Gamma, F, G \rhd G$ or $\Gamma, F \rhd F$. For both of these states, **C** has a winning strategy by (1). □

Proposition 11 *If* $\boldsymbol{C}$ *has a winning strategy in* $\Gamma, (F;G), \Delta \Rightarrow H$*, then she also has a winning strategy in* $\Gamma, G, F, \Delta \Rightarrow H$.

Proof. Let τ be a winning strategy for **C** in $\Gamma, (F;G), \Delta \Rightarrow H$. **C** can use essentially the same strategy τ in $\Gamma, G, F, \Delta \Rightarrow H$. If during the game, the indicated occurence of G, F is on top of the stack and τ tells her to (Check $(F;G)$), **C** simply skips this step. □

The converse to Proposition 11 fails: For example, **C** has a winning strategy in

$$K, F \to G, G \to H \rhd F \to H$$

as the following **LSTACK**-derivation shows:

$$\dfrac{\dfrac{K, F, F \to G, H \Rightarrow H \qquad \dfrac{K, F, G \Rightarrow G \qquad K, F \Rightarrow F}{K, F, F \to G \Rightarrow G}\,(\to\mathrm{L})}{K, F, F \to G, G \to H \Rightarrow H}\,(\to\mathrm{L})}{K, F \to G, G \to H \Rightarrow F \to H}\,(\to\mathrm{R})$$

In contrast, **C** has no winning strategy in $((F \to G); K), G \to H \rhd F \to H$. This is because $(;)$ prevents **C** from inserting the premise F below $F \to G$ in the stack as her first step in the winning strategy. One easily checks that no other proof exists, assuming that F, G, H, K are pairwise distinct atoms.

The discussed properties allow one to wrap up whole game states in single information packages: For any game state $S \equiv F_1, \ldots, F_n \rhd G$ let $\mathit{IP}(S) := ((\ldots(F_n; F_{n-1}); F_{n-2}); \ldots); F_1) \to G$.

Proposition 12 *C has a winning strategy in a game state S iff C has a winning strategy in the state $\rhd \mathit{IP}(S)$.*

Proof. For the direction from left to right, **C** starts the game for $\rhd \mathit{IP}(S)$ by sending a (Check $\to$)-request, followed by $(n-1)$-many Unpack$(;)$-requests. The game is then in the state S, for which **C** has a winning strategy by assumption. For the other direction, it is clear (by lack of other choices) that a winning strategy for $\mathit{IP}(S)$ must start with a (Check $\to$)-request, and hence **C** has a winning strategy for the subsequent state $((\ldots(F_n; F_{n-1}); F_{n-2}); \ldots); F_1) \rhd G$. By applying Proposition 11 $(n-1)$-times, we see that **C** has a winning strategy in $F_1, \ldots, F_n \rhd G$. □

Formulated proof-theoretically, Proposition 12 says that **LSTACK** is an *internal calculus*: There is a uniform way of mapping sequents S to formulas $\mathit{IP}(S)$ such that S is provable iff its formula interpretation $\mathit{IP}(S)$ is provable.

Finally observe that combining winning strategies for different game states in **C**/**S**(**STACK**) would require to merge stacks. Hence the following observation should not come as a surprise.

Proposition 13 *The cut rule is not admissible in* **LSTACK**.

Proof. Let a, b, c be pairwise distinct atoms and $a \neq \bot$. The sequents $a, b \to c \Rightarrow b \to c$ and $b, b \to c \Rightarrow c$ are provable. Applying the cut rule (with cut formula $b \to c$) yields the sequent $b, a, b \to c \Rightarrow c$, which is not provable:

$$\dfrac{b, a, c \Rightarrow c \qquad \dfrac{}{b, a \Rightarrow b}\,??}{b, a, b \to c \Rightarrow c}\,(\to\mathrm{L})$$

□

7 Conclusion

We have introduced an interpretation of single-conclusioned sequent calculi as means of information extraction: formulas are seen as information packages and a derivation of $\Gamma \Rightarrow F$ corresponds to a winning strategy of a Client **C** that seeks to reduce the information F to the information Γ provided by the Server **S**. In

this manner we obtain an interpretation of a standard sequent calculus for intuitionistic logic that naturally extends to (affine) intuitionistic linear logic **IAL**. In particular exponentials and subexponentials receive a robust interpretation in terms of safeness from destruction through consumption. To demonstrate that our game semantics does not only fit already known calculi, we also applied it to a new concept: sequents where the left hand side represents a stack, rather than a set, multiset, or list of information packages.

We view the presented ideas and results as just a starting point for a more thorough analysis of deduction in analytic calculi in terms of reducing structured information to atomic information and plan to address, e.g., the following questions in future research: Which further operators for packaging information should be considered? Which alternative forms of storing information on a server lead to sequent calculi? Can the approach be lifted to quantifiers? Does the new interpretation of rule-admissibility lead to further insights into the underlying logics? How can the Client/Server view assist in organizing efficient proof search?

References

1. Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *The Journal of Symbolic Logic*, 59(02):543–574, 1994.
2. Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. Logic Comput.*, 2(3):297–347, 1992.
3. Andreas Blass. A game semantics for linear logic. *Annals of Pure and Applied logic*, 56(1):183–220, 1992.
4. Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx. The structure of exponentials: Uncovering the dynamics of linear logic proofs. In *Kurt Gödel Colloquium on Computational Logic and Proof Theory*, pages 159–171. Springer, 1993.
5. Jean-Yves Girard. Linear logic: its syntax and semantics. In *Advances in linear logic (Ithaca, NY, 1993)*, volume 222 of *London Math. Soc. Lecture Note Ser.*, pages 1–42. Cambridge Univ. Press, Cambridge, 1995.
6. Giorgi Japaridze. The intuitionistic fragment of computability logic at the propositional level. *Annals of Pure and Applied Logic*, 147(3):187–227, 2007.
7. Laurent Keiff. Dialogical logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2011 edition, 2011.
8. Yves Lafont and Thomas Streicher. Games semantics for linear logic. In *LICS'91. Proceedings of Sixth Annual IEEE Symposium on Logic in Computer Science*, pages 43–50. IEEE, 1991.
9. Paul Lorenzen. Logik und Agon. In *Atti del XII Congresso Internazionale di Filosofia*, volume 4, pages 187–194, 1960.
10. Francesco Paoli. *Substructural logics: a primer*, volume 13. Springer Science & Business Media, 2013.
11. Shahid Rahman and Helge Rückert. Dialogical connexive logic. *Synthese*, 127(1):105–139, 2001.
12. A. S. Troelstra and H. Schwichtenberg. *Basic proof theory*, volume 43 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, second edition, 2000.