STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVII**, Number 4, 2012

NETWORK ROUTING MODELLED BY GAME SEMANTICS

DANIEL MIHÁLYI, VALERIE NOVITZKÁ, PETER PRAZŇÁK,
AND PETER POPOVEC

Abstract. An important task in network routing is to find the best path for message passing through the computer networks. In our paper we show how network routing can be described by linear logic formulae and modeled as a game tree in terms of game semantics.

1. Introduction

One of the primary needs of routing in computer networks is to state the best path for delivering a message from a sender to a receiver with respect to a given network topology. Presently, information about available paths are dynamically actualized in every router and it is stored in dynamic routing tables. The best way is determined according to an actual situation in a network by executing appropriate algorithm returning a value of the least metrics computed from path cost, channel capacity and speed, eventually response time and other parameters. From the computer network's point of view, a choice of the best path is a mostly deterministic process depending on an actual network environment. Hovewer, a user does not have and he does not need any information about the environment in which his message is passed to a receiver, and the best path choice appears him as a non-deterministic process.

The aim of our paper is to propose a formal description of network routing using an appropriate logical system in such a way that it reflects both the user's and enviroment's point of view, respectively. Then we construct a model that is illustrative enough for the best path choice.

For the description of network routing, we choose the language of linear logic introduced by J. Y. Girard in [11]. In contrast to classical logical systems, linear logic can express the dynamics of processes, sequentiality of particular actions, and eventually, resource manipulation, namely space and time on

Received by the editors: June 10, 2012.

2010 *Mathematics Subject Classification.* 03F52, 91A40.

1998 *CR Categories and Descriptors.* F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic – *Computational logic*; I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems – *Games*.

Key words and phrases. game semantics, linear logic, network routing.

logical level. Linear logic has many applications in computer science. The processes described by formulae of linear logic can be modeled by Petri nets [18], Turing machines, special categories [21] or game semantics. There exist several logic programming languages based on linear logic, e.g. Lygon [15], LPP [4, 19], etc.

In our approach we exploit the special property of additive conjunction that can be understood variously from different points of view. The author of linear logic interpreted additive conjunction as a choice between alternative actions conditioned by an environment. Girault in his work [14] sees the sense of additive conjunction as a deterministic selection, i.e. a kind of deterministic choice in a situation, where several actions are available but only one of them can be performed. In our approach, we interpret this conjunction as a synthesis of both previous views for the designation of the best path. This logical connective expresses non-deterministic choice from the user's point of view and dependent choice from the network's point of view arising from actual environment.

The semantics of linear logic was formulated by various models: coherent spaces, phase spaces [12], symmetric monoidal closed categories [3], game semantics, etc. For our approach, we use the game semantics method that models linear formulae as polarized game trees.

Game semantics is based on the game theory formulated by John Von Neumann and Oscar Morgensen in 1944. Game theory [20] is a mathematical discipline enabling to model real situations by games and it is used mainly in economics. It helps to solve decision problems and to search winning strategies. Originally, game semantics was formulated by P. Lorentzen [17] for the justification of intuitionistic logic. Game semantics was formulated in 90-ies of 20. century and has found many applications in computer science. For instance, modeling of interactions [1], defining semantics of various logical systems and programming languages [8, 7], modeling and verification of software [2, 9], in hardware design [10], in linguistics and artificial intelligence.

Game semantics for linear logic was formulated and published by many authors, e.g. [1, 2, 5, 6, 8]. A structure used as a model of this method is a game tree which is a suitable structure for representing behavior of computer networks. A game tree is a directed graph without loops and its firm enables us to see the posssible winning strategy in finding the best path from a source to a destination.

In Section 2, we present a short overview of linear logic and we introduce the syntax of the fragment of linear logic suitable for our purposes. Section 3 introduces the basic notions and principles of game semantics together with an interpretation of linear logic connectives. In Section 4, we show how linear

logic can be used for describing network routing by linear formulae and how its game semantics can serve for modeling network routing by game tree.

2. Linear logic

Logical systems play important role in computer science. Classical logics (propositional and predicate logics) have their well-established applications within description and proving of some program properties. They are often used in specification languages for stepwise refinement of specifications to implementation, in program verification and many other areas of computer science.

Non-classical logic, for instance temporal logic enables to state the truth of formulae depending on time. Further modal logics enrich proposition or predicate logics with the operators of the necessity and possibility (classical modal logic), knowledge and belief (epistemic logic), obligation, permission and forbiddance (deontic logic). None of these logical systems does not be able to describe dynamics of processes running in real world already on syntactic level. The first logical system enabling to describe dynamics of processes, their causality and resource handling is linear logic formulated by J. Y. Girard in 1987 [11, 12]. The subformulae of the classical logic's formulae may be either true or false. This property is statical, the truth value of a subformula is stable in the frame of whole formula. Therefore we can say that classical logic has statical nature, it is suitable for description of fixed situations, but not for the description of dynamics in real world processes. The formulae of linear logic can be regarded as resources that can be produced and consumed. This fact can be expressed by linear implication

$$\varphi \multimap \psi \tag{1}$$

where a resource φ is consumed after performing linear implication, which can be denoted as linear negation $\varphi^{\perp}$. A resource ψ is produced by performing linear implication (1). For instance, if we would like to travel by a train we must to have some amount of money φ for buying a ticket ψ. After an execution of this process $\varphi \multimap \psi$, our amount of money is consumed, $\varphi^{\perp}$, but we obtain a new resource, a ticket ψ.

Linear formulae can be regarded also as actions. In this case linear implication expresses causality. An action φ is a cause of an action ψ.

Linear logic introduces new logical connectives. In this paper we use a fragment of linear logic with the following syntax:

$$\varphi ::= \mathbf{0} \mid \mathbf{1} \mid \top \mid \bot \mid p \mid \varphi \otimes \psi \mid \varphi \multimap \psi \mid \varphi \,\&\, \psi \mid \varphi \oplus \psi \mid \varphi \parr \psi \mid \varphi^{\perp} \tag{2}$$

where

- p denotes atomic actions without internal structure;
- $\varphi \otimes \psi$ is multiplicative conjunction expressing that both actions φ and ψ are performed. The neutral element of this logical operation is the constant $\mathbf{1}$;
- $\varphi \multimap \psi$ is linear implication where action φ is a cause of an action ψ;
- $\varphi \ \& \ \psi$ is additive conjunction expressing external non-determinism. Only one action is executed depending on environment. The neutral element of this logical operation is the constant $\top$;
- $\varphi \ \oplus \ \psi$ is additive disjunction expressing also the execution of one action, but we do not determine which one will be performed. Additive disjunction expresses internal non-determinism, it is a dual logical operation to additive conjunction. The neutral element is the constant $\mathbf{0}$;
- $\varphi \parr \psi$ is multiplicative disjunction expressing: if φ is not performed then ψ is performed and vice versa;
- $\varphi^{\perp}$ denotes linear negation, it expresses that after performing an action φ the reaction $\varphi^{\perp}$ arises. Linear negation is involutive i.e. $\varphi^{\perp\perp} \equiv \varphi$.

3. Game semantics

In this section we formulate the semantics of our fragment of linear logic in terms of game semantics. We use dialogue games where two players participate: a proponent and an opponent. The notion game denotes a collection of rules how to play it. An actual performing of a game according these rules we call a game session. The aim of a game session is to find winning strategy i.e. a sequence of moves of proponent and opponent leading to success. Sometimes, a move of a proponent or an opponent is called a half move, if we are interesting only in winning strategy for a proponent. In the case of linear logic a proponent efforts to prove the truth of formula, an opponent efforts to deny the validity of this formula. A winning strategy is a sequence of moves leading to the success of proponent, i.e. to proving formula's validity.

Game semantic provides very useful graphical representation of a game session called game tree. A game tree is a pair

$$T = (V, A) \tag{3}$$

i.e. a directed graph consisting of a set of vertices V and a set of arrows A. A game tree for our fragment of linear logic is a digraph without loops. The vertices represent the positions in a game session and the arrows represent the moves of proponent and opponent. A game session is represented by a path from the root to the leaves in a game tree. We are looking for winning

strategy, i.e. a path in game tree, which leads to the win of a proponent. The root of the game tree is the starting point of a game session. The leaves contain information about its success or non-success.

We formulate the game semantics of our fragment of linear logic as follows: every logical connective can be translated to a fragment of a game tree.

Additive conjunction $\varphi \,\&\, \psi$ expresses dependent choice that can be interpreted as a half move of a proponent drawn by the dash lines in Figure 1. The vertice $\varphi \,\&\, \psi$ has two sons reflecting the situation that only one of the actions φ and ψ will be executed. A decision, which of them will be performed depends on proponent or implies from the environment.

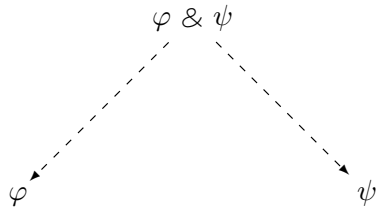

FIGURE 1. Interpretation of additive conjunction

In the case of additive disjunction a process can follow also only with one action but a proponent cannot predict with which one. Additive disjunction can be translated to the fragment of a game tree in Figure 2.

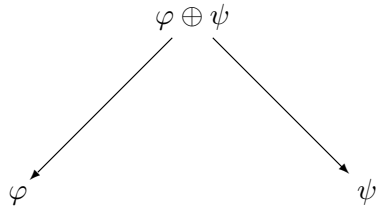

FIGURE 2. Interpretation of additive disjunction

To translate multiplicative conjunction $\varphi \otimes \psi$ to a fragment of a game tree is harder. Multiplicative conjunction expresses that both actions φ and ψ will be performed. Which of them will be performed as the first depends on a proponent's decision. Figure 3 reflects this idea.

Multiplicative disjunction $\varphi \parr \psi$ expresses a situation: if φ is not performed then ψ is performed or if ψ is not performed then φ is. A fragment of a game tree for multiplicative disjunction is in Figure 4.

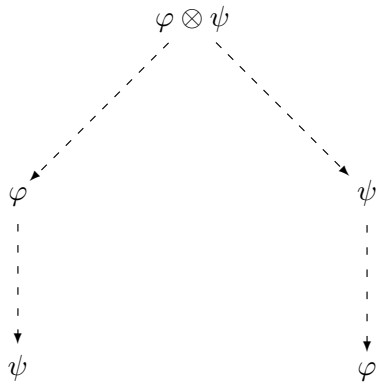

FIGURE 3. Interpretation of multiplicative conjunction

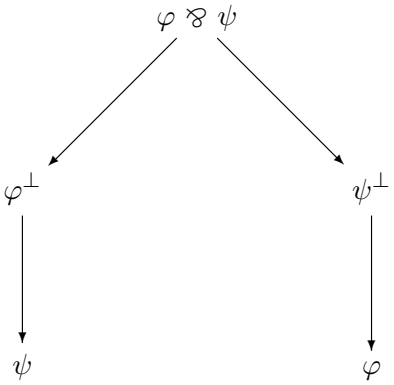

FIGURE 4. Interpretation of multiplicative disjunction

Linear implication $\varphi \multimap \psi$ expresses causality, i.e. an action φ is a cause of an action ψ. Following this idea, linear implication can be translated to the fragment of game tree in Figure 5.

FIGURE 5. Interpretation of linear implication

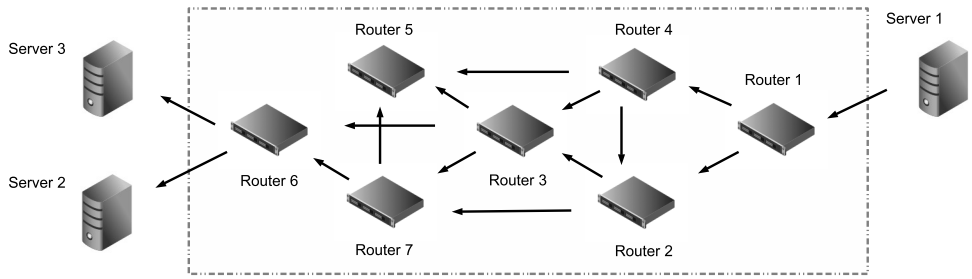

FIGURE 6. Sending a message trough a network

In the syntax of our fragment of linear logic we do not consider exponentials. The exponentials $!\varphi$ and $?\varphi$ enable to express inexhaustibility and potential inexhaustibility, respectively. Translation of these operators introduces loops into the game trees which become directed graphs with loops and they are not suitable for our approach.

4. Example: Network routing

In this section we present a simple example of network routing. We show how this problem can be formulated by linear formulae and modeled by game semantics.

We assume a network in Figure 6 consisting of seven routers denoted by $Router1, Router2, \ldots, Router7$ and three servers $Server1, Server2$ and $Server3$. We consider a case that $Server1$ sends a message trough this network to $Server2$. We are interesting in all possible paths from a source server to a target server. This situation is resolved at 3rd layer of OSI model - at network layer - where routing protocols are defined. Information about the possible paths in a network are stored in the routing tables that can be

- static routing tables or
- dynamic routing tables.

If the static routing tables are used, a message follows the path stated explicitly by network administrator. In a case of dynamic routing tables, the passing paths are actualized depending on an interconnection state between the adjacent routers. If some path is not accessible either of overloading or connection interruption, one of the other accessible paths is selected.

In the first step, we specify this problem using linear logic. We denote by

- $p_i, i = 1, \ldots, 7$ express the i-th router in our network;
- $p_i \multimap p_j$ is linear implication expressing that a router p_i sends the message to the router p_j directly, i.e. p_i and p_j are neighbouring and interconnected;
- assuming dynamic routing tables, linear additive conjunction $p_i \,\&\, p_j$ expresses that a message proceeds either to the router p_i or p_j. That means both routers are accessible but a user cannot predict which path will be used. Form the network's environment a choice is made depending on a value of metrics.

From Figure 6 we see that several accessible paths through this network are possible. A message enters into a network through the router p_1 and outputs from this network through the router p_6. From the router p_1 a message can follow either to p_2 or to p_4. This move we can specify by a linear formula

$$p_1 \multimap (\, p_2 \,\&\, p_4 \,). \tag{4}$$

Similarly, we specify how a message can follow from any router in our network. The particular steps we can describe by the following linear formulae:

$$\begin{array}{l} p_2 \multimap (\, p_3 \,\&\, p_7 \,) \\ p_3 \multimap (\, p_5 \,\&\, p_6 \,\&\, p_7 \,) \\ p_4 \multimap (\, p_2 \,\&\, p_3 \,\&\, p_5 \,) \\ p_7 \multimap (\, p_5 \,\&\, p_6 \,) \end{array} \tag{5}$$

An accessible path in our network can be specified by linear implication, for instance

$$p_1 \multimap p_4 \multimap p_3 \multimap p_7 \multimap p_6 \tag{6}$$

We see that from the router p_5 there is no passing path to p_6, therefore we have no linear formula starting from p_5. For p_6 we cannot formulate linear implication, because a message leaves a network through this router and follows to the given server. From Figure 4 it is straightforward that every path ending in p_5 is dead end.

Using translation rules introduced in previous section, for every formula in (4) and (5) we can construct a game tree in Figure 7. The root of this game tree is an antecedent of linear implication p_1 with one son, the succedent of implication $p_2 \& p_4$. This vertice has two sons p_2 and p_4 representing dependent choice. Applying this consecution for every vertice we get a game tree where leaves are either p_6 or $\Box$. The path ending with p_6 is passing path, a sended message successfully passes through our network. The paths in game tree ending with $\Box$ indicate dead ends, i.e. these paths cannot be used for transporting

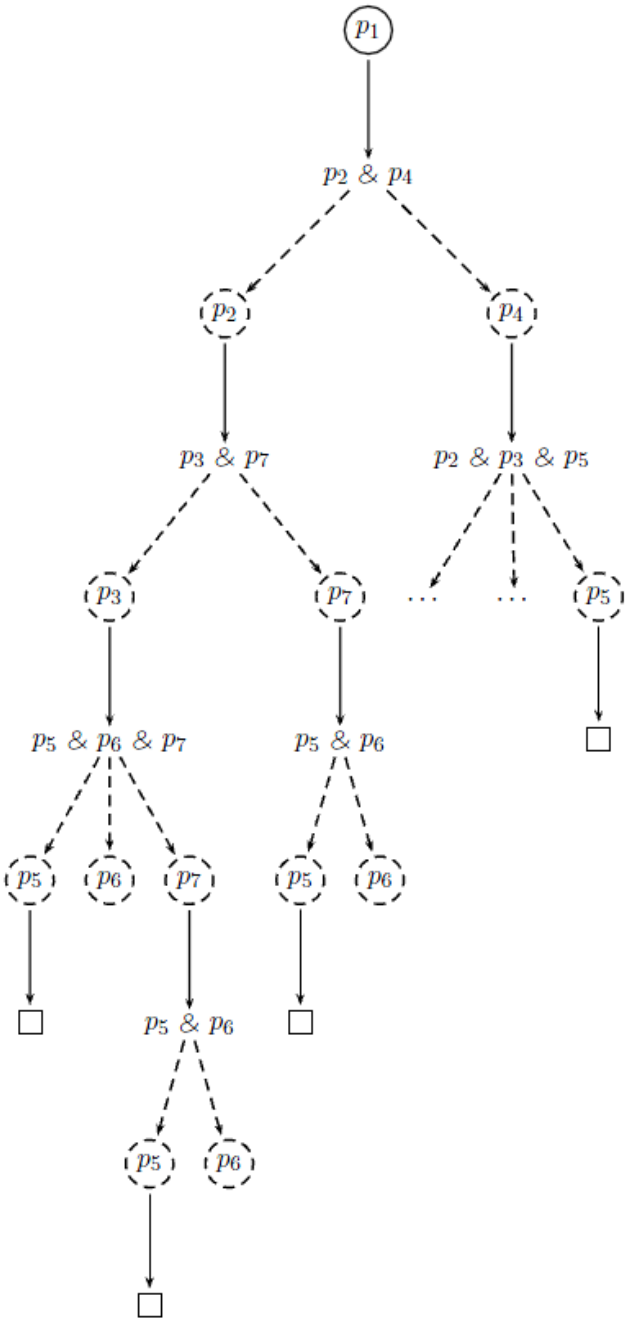

FIGURE 7. Game tree for network routing

a message through our network successfully. If we simplify a value of metrics as a number of hops, we obtain this information from a path's deep in a game tree. From Figure 7 we see that the shortest path is $p_1 \multimap p_2 \multimap p_3 \multimap p_6$ and it is the best path for routing in our network.

5. Conclusion

In our paper we present an illustration how network routing can be described by formulae of linear logic and modeled by game semantics. We define interpretation of linear connectives by the fragments of game tree. Our approach seems to be appropriate to specify dependent choice of the best network path from a game tree.

In our further research we would like to extend our approach by including exponentials to our fragment of linear logic expressing that some actions can be potentially used repetitiously. Another extension cane be done by labeling moves by metrics values computed by some version of Dijkstra algorithm. We hope that our approach enriched with exponentials can be suitable for specifying and modeling reliable program systems.

6. Acknowledgement

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-0008-10 "Modelling, simulation and implementation of GPGPU-enabled architectures of high-throughput network security tools."

This work is the result of the project implementation: Center of Information and Communication Technologies for Knowledge Systems (ITMS project code: 26220120030) supported by the Research & Development Operational Program funded by the ERDF.

References

[1] Abramsky, S.: Semantics of Interaction: An Introduction to Game Semantics, In: Proceedings of the 1996 CLiCS Summer School, Isaac Newton Institute, P. Dybjer and A. Pitts, eds. (Cambridge University Press) 1997, pp.1-31

[2] Abramsky, S., Ghica, D., Ong, L., Murawski, A.: Applying Game Semantics to Compositional Software Verification, Proc. of the 10th Intern.Conf. Tools and Algorithms for the Construction and Analysis of Systems, Springer LNCS 2988, 2004, pp.421-435

[3] Ambler, S.,J.: First Order Linear Logic in Symmetric Monoidal Closed Categories, Research Report ECS-LFCS-92-194, University of Edinburgh, 1992

[4] Banbara M., Tamura N.: Compiling Resources in a Linear logic Programming Language, Proceedings of JICSLP'98 Implementation Technologies for Programming Languages Based on Logics, June 1998, pp. 32–45,

[5] Blass A.: Is Game Semantics Necessary?, In (Boerger, E., Gurevich, Y. and Meinke, K. eds.): 7th. Workshop Computer Science Logic, CSL'93, Springer LNCS 832, 1994, pp. 66–77,
[6] Blass A.: A Game Semantics for Linear logic, Annals of pure and Applied Logic, Vol 56, 1992, pp. 182–220,
[7] Curien P.-L.: Notes on Game Semantics, Techn.Rep.CNRS University of Paris, 2006
[8] Delande, O.: Symmetric Dialogue Games in the Proof Theory of Linear Logic, PhD. Thesis, École Politechnique 2009,
[9] Dimovski, A., Lazic, R.: Assume-Guarantee Software Verification Based on Game Semantics, Proc. on Formal Methods and Software Engineering, Springer LNCS 4260, Macao, 2006, pp. 529–548,
[10] Ghica D.,R.: Application of Game Semantics: From Program Analysis to Hardware Synthesis, School of Computer Science, University of Birmingham, 2009,
[11] Girard, J.-Y. : Linear Logic, Theoretical Computer Science, London Mathematical 50:1, pp. 1-102, 1987.
[12] Girard, J-Y., Lafont, Y.,Taylor, P.: Proof and Types, Cambridge University Press (Cambridge Tracts in Theoretical Computer Science, 7), 1990
[13] Girard, J-Y.: Linear logic: Its Syntax and Semantics, Cambridge University Press, 2003
[14] Girault C., Valk R.: Petri nets for systems engineering - a guide to modeling, verification, and applications, Springer, ISBN 3-540-41217-4, 2003, pp. 370–382,
[15] Harland J., Pyms D., Winnikof M.: Programming in Lygon and Overview, Algebraic Methodology and Software Technology, Springer 1996,
[16] Lafont, Y.,Streicher, T.: Games Semantics for Linear Logic, Logic in Computer Science (LICS 91), p. 43-50, IEEE Computer Society Press, 1991,
[17] Lorentzen P.: A Dialogue Criterium for Constructivity, In Infinistic methods, Warszaw, 1961, pp. 193–200
[18] Mihályi D., Novitzká V., Slodičák V.: From Petri Nets to Linear Logic, CSE'2008, Vysoké Tatry - Stará Lesná, 24. - 26. 9. 2008, Košice, Elfa, 2008, pp. 48-56, ISBN 978-80-8086-092-9,
[19] Miller D.: Overview of Linear Logic Programming, Linear Logic in Computer Science, Cambridge University Press, 2000,
[20] Peters, H.: Game theory - a Multi-leveled Approach, Springer-Verlag, 2008
[21] Slodičák V.: Some Useful Structures for Categorical Approach for Program Behavior, Journal of Information and Organisation Sciences, Vol. 35, No. 1, 2011, pp. 93–103

Technical University of Košice
E-mail address: (Daniel.Mihalyi,Valerie.Novitzka)@tuke.sk, praznak@minv.sk,, popovec@fei.tuke.sk