A Study on Completeness and Soundness of the Connection Graph Proof Procedure

川　口　雄　一
Yuuichi KAWAGUCHI

In 1975, R. A. Kowalski introduced the connection graph proof procedure, and in 1981 W. Bibel proved that it is complete and sound. The purpose of this paper is to show concrete examples of completeness and soundness in propositional logic. Two examples are given. One is an example of completeness, and the other is an example of soundness. The reason why they are concrete examples is discussed. When those two examples are generalised, then it is proven that the connection graph proof procedure is complete and sound. However, the proof is not shown in this paper.

Key words: completeness, soundness, connection graph, propositional logic.

天使大学 看護栄養学部 教養教育科

(2006年１月20日受稿、2006年４月４日審査終了受理)

1. Introduction

In general, completeness and soundness (or *consistency* in [1]) of a proof procedure for arguments are defined as follows.

Completeness: If an argument is valid, then it is proven by the proof procedure.

Soundness: If an argument is proven by the proof procedure, then it is valid.

An argument consists of some assumptions and the conclusion. Assumptions imply the conclusion, if and only if the argument is *valid*. An argument not being valid is *invalid*.

In 1975, R. A. Kowalski introduced the *connection graph proof procedure* [6] in first-order predicate logic. In 1981, W. Bibel proved [1] that the connection graph proof procedure is complete and sound. The logical expressions in that paper do not have variables and substitutions. The proof is given in propositional logic.

The purpose is to give concrete examples of completeness and soundness of the connection graph proof procedure in propositional logic. Two examples are given. They are concrete examples of completeness and soundness. When two examples are generalised, then it is proven that the connection graph proof procedure is complete and sound. However, the proof is not shown in this paper.

2. Connection Graph Proof Procedure

The connection graph proof procedure [7] is a method based on resolution [3]. The procedure proves a set of clauses to be consistent or inconsistent.

2.1 Connection Graph

Logical expressions are expressed in clausal forms (or clauses). For example, a logical expression "if X, Y and Z, then S or T" is represented as a clause "$S, T \leftarrow X, Y, Z$," where S, T, X, Y and Z are propositions.

Given a set of clauses, if there is a proposition in a clause on the left side of the arrow sign '$\leftarrow$' and the same proposition exists on the right side in another clause, then those two corresponding propositions are connected by an edge.

When an argument is given, a set is made of assumptions of the argument and a negation of the conclusion. For a given set of clauses, a graph made by connecting all corresponding pairs of propositions is called a 'connection graph.'

2.2 Connection Graph Proof Procedure

A connection graph is solved if it has an empty clause "$\leftarrow$." A set of clauses is proven to be inconsistent if the connection graph made from the set is solved. An argument is valid if a set made from the argument is inconsistent.

A connection graph that does not have an empty clause can be translated into another connection graph by the following *connection graph proof procedure*:

1. Remove clauses that are tautologies from the graph.
2. Remove clauses that contain a proposition not have any corresponding proposition.
3. Select an edge from the graph, then remove it, add a *resolvent* into the graph, and connect corresponding propositions by edges.

The graph is sequentially translated into other graphs. If an empty clause is obtained, then the original set of clauses is proven to be inconsistent.

NOTE 1 [7]: A clause is a tautology if it contains the same proposition on the left and right sides of the arrow sign.

NOTE 2: For example, from two clauses

$S_1, \boxed{P} \leftarrow X_1, Y_1, Z_1$ and $S_2, T_2 \leftarrow X_2, \boxed{P}, Z_2$, a resolvent $S_1, S_2, T_2 \leftarrow X_1, Y_1, Z_1, X_2, Z_2$ is obtained.

3. Concrete Examples

There are two examples in this section. These examples are variations on an example described in Chapter 8 of the book [7]. The original one is in first-order predicate logic, and those in this section are in propositional logic.

3.1 Example One

There are two propositions:

J: John is happy.

W: Bob is working.

Let us consider an argument that consists of two assumptions and a conclusion:

Assumption 1: $J \leftarrow W$.

Assumption 2: $W \leftarrow$.

Conclusion: $J \leftarrow$.

The negation of the conclusion is '$\leftarrow J$.' A set $\{J \leftarrow W,\ W \leftarrow, \leftarrow J\}$ is obtained. The connection graph (*initial connection graph*) is shown in Figure 1.

The graph is sequentially translated into other graphs (Figure 2) in the direction of the finger-arrow sign (☞). There are five graphs in Figure 2.

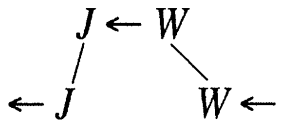

Fig. 1

At first, in the second graph, the two clauses '$J \leftarrow W$' and '$W \leftarrow$' are resolved and the edge is removed (marked with a cross), and then the resolvent '$J \leftarrow$' (surrounded by a dashed rectangle) is added and connected by an edge. Now the proposition 'W' in the clause '$W \leftarrow$' is not connected to any corresponding proposition, and the clause is then removed (surrounded by a dashed circle).

In the third graph, the proposition 'W' in the clause '$J \leftarrow W$' is also not connected. The clause and the edge are removed.

In the fourth graph, the two clauses '$J \leftarrow$' and '$\leftarrow J$' are resolved and the edge is removed, and then the resolvent '$\leftarrow$' (an empty clause) is added. Each proposition 'J' in those two clauses is now not connected, and the two clauses are removed.

Finally, in the fifth graph, an empty clause '$\leftarrow$' is obtained, and then the graph is solved. Therefore, the original set of clauses is proven to be inconsistent and the argument is valid.

3.2 Example Two

There is another clause 'P: Bob is playing.' Let us examine whether another argument is valid: two assumptions '$J \leftarrow W$' and '$P \leftarrow$' imply the conclusion '$J \leftarrow$.'

A set $\{J \leftarrow W, P \leftarrow, \leftarrow J\}$ is obtained, and the initial connection graph is shown in Figure 3.

The graph is sequentially translated into. other graphs (Figure 4). Clauses that have a proposition not connected (surrounded by a dashed circle) are sequentially removed. Consequently, there is no clause in the graph. The original set of clauses is translated into an empty set. The result shows that the set is consistent and the argument is invalid. This is described in Section 4.2.

4. Discussion

4.1 Example One

Example one shows that the connection graph proof procedure proves an argument named "*syllogism*" (*i.e.*, '$J \leftarrow W$' and '$W \leftarrow$' imply '$J \leftarrow$'). The syllogism is valid, which is proved (*e.g.*, by Tableau [5],[10]).

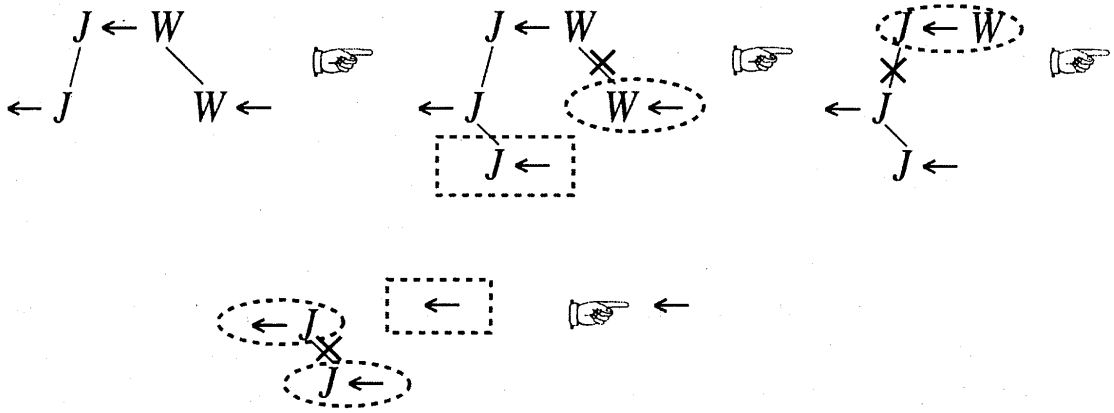

Fig. 2

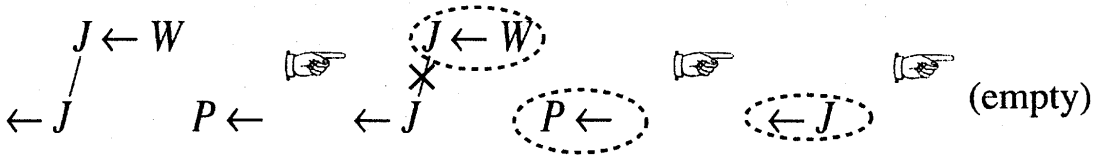

Fig. 4

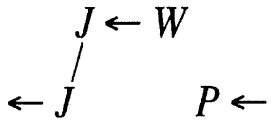

Fig. 3

In the example, three clauses that construct the argument are given. A set of assumptions and a negation of the conclusion is represented in an initial connection graph. The graph is sequentially translated into other graphs, and finally a graph including an empty clause is obtained. The graph is solved and the set of clauses is proven to be inconsistent. Thus, the argument is valid.

Therefore, this is an example in which if an argument is valid, then an empty clause is obtained. The example one is a concrete example of completeness.

4.2 Example Two

Example two shows that if an argument is invalid, then a set of assumptions and a negation of the conclusion is translated into an empty set.

The soundness of the connection graph proof procedure is that if an empty clause is obtained from the original set of clauses, then the argument corresponding to the set is valid. The contrapositive proposition of this is that if an argument is invalid, then an empty clause is not obtained.

In the example, the argument is invalid, which is proven (e.g., by Tableau).

In a situation in which an original set of clauses is translated into an empty set, the set is proven to be consistent by "affirmation soundness." In another situation in which an empty clause is obtained, the original set is inconsistent by completeness. A set can not be consistent and inconsistent at the same time. Only one of those situations can be. In the example, the set is translated into an

empty set. Thus, an empty clause is not obtained.

Therefore, this is an example in which if an argument is invalid, then an employ clause is not obtained. The example two is a concrete example of soundness.

NOTE 3 (*affirmation soundness* [8]): If a set of clauses is translated into an empty set by the connection graph proof procedure, then the set is satisfiable (⇔ consistent).

5. Conclusion

There are two examples in this paper. Example one is a concrete example of completeness of the connection graph proof procedure. Example two is a concrete example of soundness.

When two examples are generalised, then we obtain a proof of completeness and soundness of connection graph proof procedure. However, the proof is not shown in this paper.

Furthermore, some researchers have published articles (*e.g.*, [2], [4] and [8]) showing that there is some doubt about strong completeness of the connection graph proof procedure. Strong completeness is not discussed in this paper.

Acknowledgements

This study was supported by a grant from Tenshi College (Title: "*A Development of On-line Materials for Teaching Subjects Concerning Statistics*").

For mathematical definitions, the author referred mainly to the dictionary [9] edited by the Mathematical Society of Japan. For technical terms and phrases in English, the author referred mainly to R. A. Kowalski's book [7] and the dictionary [11] edited by Y. Komatsu.

The author is grateful to the anonymous reviewer for constructive suggestions, which helped to improve the clarity of this paper.

References

[1] Wolfgang Bibel. On matrices with connections. *Journal of the ACM*, 28(4) pp. 633-645, 1981.

[2] Wolfgang Bibel. Decomposition of tautologies into regular formulas and strong completeness of connectiongraph resolution. *Journal of the ACM*, 44(2) pp. 320-344, March 1997.

[3] Jean H. Gallier. *Logic for Computer Science—Foundations of Automatic Theorem Proving.* Harper & Row, 1986.

[4] Reiner Hahnle, Neil V. Murray, and Erik Rosenthal. Ordered resolution vs. connection graph resolution. In *Proceedings International Conference on Automated Reasoning (IJCAR 2001)*, Siena, Italy, 2001. Springer-Verlag LNAI, (CiteSeer #446349).

[5] Richard Jeffrey. *Formal Logic: Its Scope and Limits.* McGraw-Hill, Inc., third edition, 1991.

[6] Robert A. Kowalski. A proof procedure using connection graphs. *Journal of the ACM (JACM)*, 22(4) pp. 572-595, 1975.

[7] Robert A. Kowalski. *Logic for Problem Solving.* Elsevier Science Publishing Co., Inc., New York, 1979, (translated into Japanese).

[8] Jörg Siekmann and Graham Wrightson. An open research problem: Strong completeness of R. Kowalski's connection graph proof procedure. *Lecture Notes in Artificial Intelligence (LNAI)*, Vol. 2408, pp. 231-252, August 2002.

[9] 日本数学会（編）. 岩波数学辞典. 岩波書店, 東京, 第3版, 1985年12月.

[10] 丹治信春. タブローの方法による論理学入門. 朝倉書店, 東京, 1999年11月.

[11] 小松勇作（編）. 数学英和・和英辞典.共立出版, 東京, 1979年7月.