Fórmulas con signo

Formulas with sign

*Nelson Becerra Correa

**Edgar Altamirano Carmona

Fecha de recepción: 17 de octubre de 2007
Fecha de aceptación: 20 de febrero de 2008

Resumen

En este artículo, se hace una descripción breve del problema SAT signado (*SAT Signed*). Las fórmulas signadas son un lenguaje para la representación del conocimiento [1], que se encuentra ubicado en la intersección de propagación de restricciones (CP), Lógica multivaluada (MVL) y programación lógica annotated (ALP).

Una fórmula normal conjuntiva con signo (FNC Signada) está compuesta por; la conjunción de cláusulas disyuntivas con signo. Cada cláusula contiene literales con signo. Una literal, es llamada un átomo signado, el cual es una expresión de la forma $S{:}p$, donde p es un átomo clásico y S es su signo, el cual se compone de un subconjunto de dominio N.

Las aplicaciones para deducir en lógica con signo se derivan de la programación lógica anotada (eje, bases de datos deductivas), satisfacción de restricciones (eje, problema de scheduling) y lógica multivaluada (eje, procesamiento del lenguaje natural). El papel fundamental de esta lógica justifica el estudio de algoritmos y problemas SAT asociados.

* Ingeniero de Sistemas. Magíster en Ingeniería de Sistemas, Universidad Nacional de Colombia. Magíster en Inteligencia Artificial, Universidad Politécnica de Cataluña. DEA en Inteligencia Artificial. Docente de la Universidad Distrital Francisco José de Caldas, adscrito a la Facultad Tecnológica. Correo electrónico: nelson@iiia.csic.es

** Profesor-investigador en la Universidad Autónoma de Guerrero. México. Áreas de interés: mundos virtuales en-linea. Moodle. Tic. Clases a distancia. Lengua nahuatl. Doctor en Inteligencia artificial, IIIA-Barcelona España. Correo electrónico: edgar@altamirano.biz

Abstract

A short description about the signed SAT is made in this article. The signed formulas are a language for knowledge representation. [1], that is found on the intersection of the restrictions of propagation. (CP-), multi-valuable logic, -(MVL-), and annotated logic programming.

A normal connective formula signed (FNC singed), is made from the conjunction of disjunctives clauses with a sign. Each clause has literals with sign. A literal is called an antonym signed. An antonym signed; which is an expression of the formula S:p, where p is a classic antonym and S is its sign, which is composed of a subgroup of N domain.

Applications to deduct a sign in logic are derived from annotated programming logic (e.g. deductive databases), satisfaction of restrictions (e.g. scheduling problem) and milti-valuable logic (e.g. processing of the natural language).

The main purpose of this logic is to give an explanation of the study of algorithms and problems related with the SAT.

Key words: multi-valuable logic, SAT, Davis and Putnam, multi-valuable formulas, regular formula.

1. Introducción

La lógica, como disciplina matemática, ha sido desde el principio uno de los fundamentos de la informática. Por otro lado, la informática ha dado un nuevo impulso a las investigaciones en lógica, y actualmente una parte importante de stas están motivadas por problemas informáticos. En este aspecto ha intervenido no sólo la lógica clásica (bivalente), sino también las multivaluadas, temporales y modales. Desde el punto de vista de la computación, el estudio de los fragmentos proposicionales, han desempeñando un papel decisivo. En particular, la lógica proposicional multivaluada tiene muchas aplicaciones prácticas en el campo de la inteligencia artificial, tales como los sistemas expertos, el razonamiento basado en casos, sistemas multiagentes, etc.

En 1917, el matemático polaco Jan Aukasiewicz publicó los primeros trabajos sobre la lógica ternaria. La base de esta lógica es que tenemos tres estados de verdad: verdadero, falso y desconocido. Estos valores se pueden representar de muchas maneras; por ejemplo, 0 para falso, 1 para desconocido, y 2 para verdadero. Se pueden definir muchos operadores lógicos, pero los más comunes son los que se muestran en la tabla 1.

Tabla 1. Tabla de verdad

x	NOT x
0	2
1	1
2	0

Como se puede ver, en las tablas 2 y 3 (MIN y MAX), estas son generalizaciones de la AND y OR de la lógica binaria. Igualmente, cumplen las leyes de De Morgan; de hecho, se pueden extender a cualquier número de valores de verdad y las seguirán cumpliendo. Esta lógica permite el desarrollo de las *Fórmulas con signo* [1], las cuales –como ya se mencionó anteriormente– son un lenguaje para la representación del conocimiento, que está en la intersección de satisfacción de *restricciones, lógica multi-valuada, programación lógica anotada.*

Tabla 2. Min generalización de AND

0	1	2	
0	0	0	0
1	0	1	1
2	0	1	2

Tabla 3. Max generalización de OR

0	1	2	
0	0	1	2
1	1	1	2v
2	2	2	2

El artículo en su primera sección, describe los conceptos sintácticos. En la segunda parte se analiza la semántica, y se describe el procedimiento Davis, Putnam, Loveland, Logemann signado; en adelante nos referimos a él como DPLL. A través de un ejemplo se detalla su funcionamiento. En la sección final se analiza la transición de fase del problema SAT-signado.

2. Preliminares

2.1 Sintaxis

Fórmulas multivaluadas (MV-formulas) MV-fórmulas son unas fórmulas que generalizan las formas normales conjuntivas (FNC) de la lógica clásica proposicional. Se diferencian en que consideran un conjunto de valores de verdad que puede contener más de dos valores y en que se puede usar literales multivaluados que se satisfacen por más de una interpretación diferente para su variable.

Además, asumen la existencia de un orden total entre los elementos del conjunto de valores de verdad. Este orden ayuda a expresar de manera más compacta algunos literales multivaluados. Nosotros asumimos que nos es dada una fórmula con signo (*Signed FNC*), compuesta por un grupo de valores de verdad. En consecuencia, se definen los siguientes conceptos.

Definición 1: un grupo de valores de verdad N es un grupo finito {i1, i2,...,i 3}, donde la cardinalidad de N es denotada por $|N|$. Un orden parcial ≤ es asociado con *N*, el cual puede ser vació {□}

Definición 2: un signo [2] es un subconjunto S∈N de valores de verdad. Una literal signada es de la forma S: p, donde S es un signo y p es una variable proposicional. El complemento de una literal signada S: p, denotado por S: p, es (N∈S): p.

Una cláusula signada es un conjunto finito de literales signadas. Una cláusula signada conteniendo únicamente una literal signada, se denomina cláusula signada unitaria (signedunit clause). Una cláusula conteniendo exactamente dos literales es llamada una cláusula binaria signada (signed binay clause). La cláusula signada vacía se denota por □.

Una fórmula signada CNF es un conjunto de cláusulas signadas. La fórmula signada, cuya cláusula es binaria es llamada una fórmula 2-CNF signada. Las cláusulas, en una fórmula signada, están conectadas, en forma conjuntiva y las literales en una cláusula signada están conectadas en forma disyuntiva. En este documento se usa S1: p1V... V Sk para representar una cláusula signada S1: p1, ... , Sk : pk

Definición 3: la longitud de una cláusula signada denotada por |C|, es su cardinalidad (el numero de literales de la cláusula). La longitud de una fórmula T, denotada |T|, es la suma de las longitudes de sus cláusulas

Definición 4: para todo i∈**N**, i denota el signo j∈|**N**| j ≥ i y ↓ i denota el signo j∈N | j ≤ i donde ≤ es un orden parcial asociado con N.

Un signo S es regular, si este es idéntico a ↑ i o ↓ i para algún i∈N. Un literal signado S: p si regular si (a) el signo S es regular. O (b) el signo S = ¯S′ es el complemento de un signo regular ¯S′. Una cláusula signada (una fórmula signada) es una cláusula regular (una fórmula CNF regular) si todas las literales son regulares.

Ejemplo 1[1]. Sea el grupo N = 1, 2, 3, 4 Con un orden como se muestra en la figura 1. Nosotros utilizamos el orden de los números, naturales, Excepto que 1 y 2 son incompatibles. Entonces el signo ↑ 1 = 1,3,4 y ↓ 1 = 1son regulares; y ↑ 1 = 2 y ¯↓ = 3 = 1,2 son complementos del signo regular. El signo 3 y 1,4 son entre otros regulares o complemento de signos regulares.

El complemento ¯↑3 de el signo regular ↑ 3 no es regular. Porque no se puede representar como ↑ i o ↓ i, para cualquier i∈N. Así un literal regular puede tener un signo que no es regular (pero es el complemento de un signo regular solamente).

Definición 5. Un signo regular S, tiene polaridad positiva (resp negativa) si ésta es de la forma ↑ i (resp ↓ i).

Una cláusula regula es una cláusula regular de Horn, si ésta contiene al menos una literal con polaridad positiva y el signo de las otras literales, son complemento de signos con polaridad positiva. Una fórmula regular CNF es una fórmula de Horn regular si todas sus cláusulas son cláusulas regulares de Horn.

Ejemplo 2. Usando el grupo de valores de verdad del ejemplo 1, y su orden. La cláusula (1) ↑ 1: p , (2) ↑ 2 : pV ↑ 3: q y 4 : q son cláusulas de Horn. La cláusula regular ↑ 1: pV ↓ 2: q no son cláusulas de Horn, ya que dichas cláusulas contienen más de una literal con polaridad positiva. ↓ 1 = ¯↑ 2, pero ↑ 4 no es igual a ¯↑ i. Para todo i∈N, la cláusula ↓ 1: p es una cláusula de Horn; sin embargo ↓ 4: p no es una cláusula de Horn (ambas cláusulas son regulares).

Definición 6. Una literal S: p es monosignada si su signo S = i, contiene un elemento. Una cláusula signada (a fórmula signada CNF), es monosignada si todas sus literales son monosignadas.

2.2 Semántica

Una *interpretación* es una instancia que asigna a cada variable proposicional un elemento del grupo de valores de verdad. Esta interpretación (*I*) satisface a una literal signada

1 Tomado Harde The SAT Problem of Signed CNF Formulas, Bernhard Beckert, Reiner H Aahnle University of Karlsruhe, Felipe Manyá Universitat de Lleida.

S:p sii la *I(p)*∈S. ésta satisface a una cláusula si ésta satisface, al menos, a una literal signada en C; y ésta satisface a una fórmula signada T, si ésta satisface a todas las cláusulas de T.

Una fórmula CNF signada (una cláusula signada) es satisfactible si es satisfecha por, al menos, una interpretación; en otro caso es insatisfactible.

Ejemplo 3. Para expresar que (2:p1 → (3:p2, 6:p3)), empleamos esta MV-cláusula: (1:p1, 3:p1 , 3:p2 , 6:p3).

Dos fórmulas signadas (cláusula signadas) son equivalentes, si son satisfechas por la misma interpretación. Son equivalentemente satisfactibles si son entre otros ambas satisfactibles o ambas insatisfactibles.

Por definición la cláusula vacía es insatisfactible y la fórmula CNF vacía es satisfactible.

2.3 Transformación a forma clausular

Toda fórmula y un subconjunto de valores finito pueden ser transformados en tiempo polinomial, en una fórmula equivalente signada CNF (en la transformación su estructura es preservada [9]); así el problema SAT de la lógica finita-valuada es polinomialmente reducible a el problema signado SAT.

En conclusión, cada fórmula CNF signada puede ser trasladada en tiempo polinomial a una fórmula regular CNF, equivalente con un orden arbitrario, utilizando el siguiente truco: una cláusula signada conteniendo literales de la forma S:p se transforma primero en una cláusula monosignada remplazando S: p por Vi∈S i :p (usando la proposición 10). Entonces, todas las literales monosignadas son eliminadas remplazando una cláusula c=i: p V D, con tres cláusulas C1 = ↑ i: p V S: q, C2 = ↓ i: p V S: q, y C3 = D V = S: q, donde q es una nueva variable proposicional que no ocurre en T y S es un signo regular arbitrario.

Una transformación directa en forma polinomial en una fórmula equivalente regular fue dada por Sofronie-Stokkermans[14].

3. Procedimiento davis putnam loveland logemann (DPLL)

En lógica clásica muchos de los más eficientes revolvedores para SAT son una variante del procedimiento DPLL [5]. En esta sección se describen las extensiones, que son propuestas por [2], al esquema DPLL, de tal manera que pueda tratar formulas CNF signadas y regulares. Éste es un procedimiento completo, para la prueba de satisfactibilidad de esta clase de fórmulas, y ver si son buenos candidatos para implementarsen por resolvedores SAT signados.

Procedimiento DPLL signado: el procedimiento DPLL signado (*Signed-DPLL*) está basado sobre las siguientes reglas:

1 Regla del literal puro signado (*Signed one-literal rule*): dada una fórmula T, que contiene una cláusula unitaria signada (unit clause S: p),

   (a) Remueva todas las cláusulas que contengan un literal *S'*: p tales que S ⊆ *S'*;
   (b Borre todas las ocurrencias del la literal *S''*: p tales que S ∩ *S''* = ∅;
   (c) Reemplace todas las ocurrencias del literal *S'''*: p con (*S'''* ∩ S):p.

2. Reglas de partición signada (*Signed branching rule*): esta regla reduce el problema de determinar entre otros si una fórmula CNF signada T (que contiene la variable

proposicional p), es satisfactoria al problema de determinar si existe un i ∈ N, tal que T ∪ i: p es satisfactible.

Definición 7: dada una fórmula signada CNF T, que contiene una cláusula unitaria S:p, simplificamos (T, S : p), denotamos el resultado de aplicar la regla del literal único a T, usando la cláusula unitaria S : p.

El procedimiento *Signed-DPLL* se muestra en el algoritmo: 3. El procedimiento anterior, primero, aplica repetida mente la regla de propagación unitaria, hasta que no se pueda simplificar más. En ese punto aplica la regla de partición y recursivamente prueba solucionar cada subproblema $|N|$ veces.

Como estos subproblemas por construcción contienen una cláusula unitaria, entonces la regla del literal único signado puede ser aplicada ahora. El procedimiento termina cuando entre otros un subproblema es satisfactorio o todos los subproblemas son insatisfactorios.

Intuitivamente, *Signed-DPLL* construye un árbol de búsqueda usando una estrategia de primero en profundidad, el nodo raíz del árbol es etiquetado con la fórmula de entrada. Los otros nodos son etiquetados luego de la

Algorithm 1 Procedimiento Signed Davis, Putnam, Log mann, Loveland $DPLL$

```
Input: a signed CNF fórmula Γ and a truth value s
  N = i_1, ..., i_n
Output: "satisfiable" or "unsatisfiable"
begin
  {/* signed one-literal rule */}
while Γ contains a unit clause {S:p} do
  simplify(Γ, S : p)
end while
if Γ = ∅ then
  return Satisfactible fi;
end if
if □ ∈ Γ then
  return Insatisfactible fi;
end if
  { signed branching rule */}
while Γ Contains a unit clause {S:p} do
  simplify (Γ, S : p)
end while
Γ := simplify (Γ, S : p);
if Γ = ∅ then
  return(true)
end if
if □ ∈ Γ then
  return(false)
end if{/* Signed braching rule */} {Let p be a prop
  sitional variable occurring in Γ}
for J= 1 TO n do
  if Signed−DPLL(Γ∪i_j : p) = "satisfactible" the
    return "satisfactible" fi
  end if
end for
return "insatisfactible"
  {end}
```

aplicación de la regla del literal puro o de la regla de partición al nodo padre.

Si todas las cláusulas contienen la cláusula vacía, entonces la fórmula de entrada es insatisfactible. En otro caso, si almenos una de las hojas, es etiquetada con la fórmula CNF vacía, la fórmula essatisfactible.

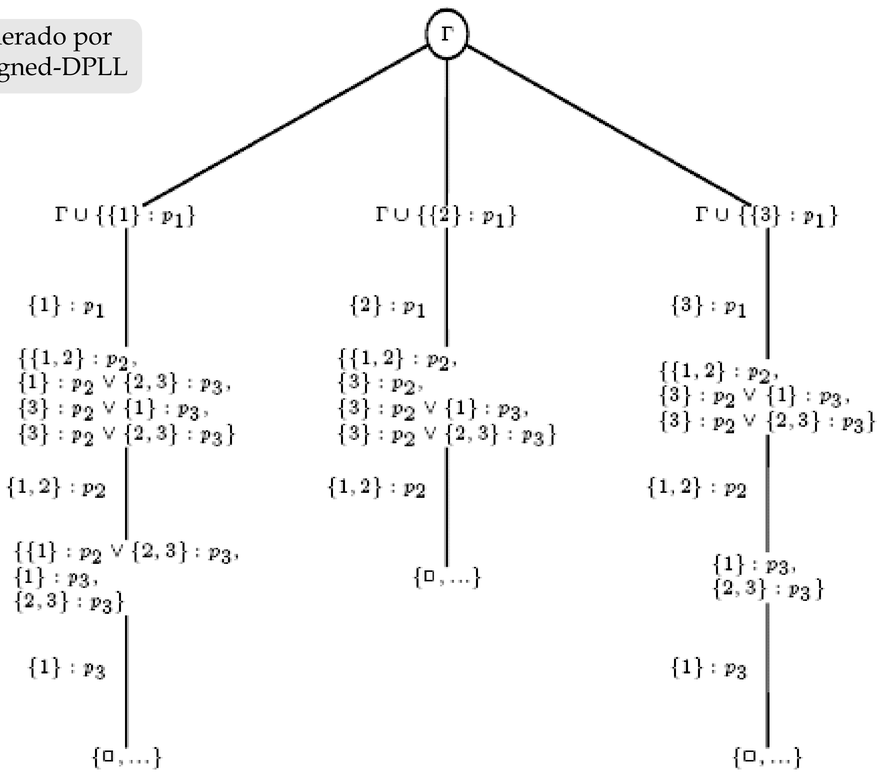
Ejemplo 4. Sea el grupo de valores de verdad *N*= {1,2,3}, con un orden arbitrario; y sea la fórmula T, con las siguientes cláusulas ver tabla 4.

La figura 1 muestra el árbol de prueba creado por *Signed-DPLL*, para una fórmula T, los nodos corresponden a la aplicación de la regla del literal puro y las etiquetas con las literales que son usadas para la simplificación.

Tabla 4. Fórmula Γ

$\underbrace{\{3\} : p_1 \vee \{1,2\} : p_2}_{C_1}$
$\underbrace{\{2,3\} : p_1 \vee \{1\} : p_2 \vee \{2,3\} : p_3}_{C_2}$
$\underbrace{\{1,3\} : p_1 \vee \{3\} : p_2}_{C_3}$
$\underbrace{\{2\} : p_1 \vee \{1,2\} : p_2}_{C_4}$
$\underbrace{\{3\} : p_2 \vee \{1\} : p_3}_{C_5}$
$\underbrace{\{3\} : p_2 \vee \{2,3\} : p_3}_{C_6}$

Figura 1. Árbol generado por el procedimiento Signed-DPLL



4. Complejidad del problema SAT-Signado

4.1 Antecedentes

Es bien conocido que el problema SAT clásico es un problema NP-Completo [13]. Sin embargo, éste es polinomial para ciertas subclases de SAT. Por ejemplo, existen algoritmos lineales, para problemas en los cuales haya como máximo una literal positiva por cláusula (Horn SAT) [6] y cláusulas, las cuales tienen como máximo dos literales (2-SAT) [12]. Similar al problema SAT clásico el problema *signed SAT* es un problema NP-Completo, pero algunas de subclases se solucionan en tiempo polinomial. Por otra parte, ya se han establecido la complejidad para Horn-Sat signado y 2-Sat signado.

Este problema tiene un grupo de valores de verdad N (*res*(N, ≥)), como un segundo parámetro de entrada.

La tabla 5 muestra una relación del problema Signed SAT y su respectiva complejidad.

Signed SAT es el problema de decidir para una fórmula arbitraria T, sobre un arbitrario grupo de valores de verdad. Si una interpretación sobre N satisface a T.

Este problema se considera un problema de decisión donde N no es un parámetro de entrada, el cual denotamos por un grupo de valores fijo. Estableciendo N valores de verdad como un índice del problema de decisión. Por ejemplo, dado un grupo de valores de verdad fijo N, signed $SATN es el problema de decidir para una fórmula arbitraria T sobre N si existe una interpretación sobre N que satisfaga a T.

4.2 Problema 2-SAT signado

El problema 2-SATN para |N| ≥ 3. Con respecto al problema 2-SAT se probo que es NP-Completo por Manyá [10] –es de recordar que el problema 2-SAT clásico es solucionado en tiempo lineal–; como una prueba alternativa de que era un problema NP-hardness de 2-SAT *signed* fue después dada por Beckert en [4]. Manyá reduce el problema 3 - colourability grafos a el problema 2-SATN para mostrar que es NP-Hard, mientras tanto Beckert [4] reduce el problema 2-SAT signed en clásico SAT.

4.3 Problema Horn-SAT regular

Un fragmento Horn está definido naturalmente (sii), el grupo de valores de verdad N esta totalmente ordenado o al menos un *retículo finito*.

Tabla 5. Complejidad *SAT Signed*

Problema	SAT	2-SAT	Horn-SAT
classical	Np-completo	lineal [12]	lineal [6]
mono-signed	NP-completo	lineal [11]	—-
classical	Np-completo	lineal [10]	lineal [8]
regular, N totally ord	Np-completo	polinomial [4]	$\|\Gamma\|log\|\Gamma\|$ [14]
regular, N a distr,lattice,signs of form $\uparrow i$ and $\bar{\uparrow} i$	Np-completo	NP-completo [4]	$\|\Gamma\|\|N^2\|$ [3]
regular, N a,lattice,signs of form $\uparrow i$ and $\bar{\uparrow} i$	Np-completo	NP-completo [10]	polinomial [4]
regular, N a,lattice,signs of form $\uparrow i$ and $\downarrow i$	Np-completo	polinomial [4]	—-
regular(arbitray)	Np-completo	NP-completo	—-
signed(arbitray)	Np-completo	NP-completo [10]	—-

Si N está totalmente ordenado, el problema de decidir entre otros si una fórmula T Horn regular es satisfactoria, puede ser solucionada en tiempo lineal en n= |T|, en el caso que |N| sea fijo y en tiempo lineal en n log n, en otro caso [8]. Otros algoritmos con la misma complejidad están descritos en [10]. Un algoritmo para una subclase particular de fórmulas Horn regular fue mencionado antes en [7].

Si N es un finito lattice, regular Horn SAT es solucionado en tiempo lineal en la longitud de la fórmula y en la cardinalidad polinomial de N vía reducción al problema Horn SAT clásico.

Referencias bibliográficas

[1] Bernhard Beckert, Reiner Hahnle, and Felip Manya. The 2-Sat Problem of Regular Signed CNF Formulas. *In Proceedings, IEEE 30th International Symposium on Multiple-Valued Logics*, pages 331-336. IEEE Press, 2000.

[2] Bernhard Beckert , Reiner Hahnle, and Felip Manyá. The SAT Problem of Signed CMF Formulas. Pages 59-80,200.

[3] Bernhard Beckert, Reiner Hahnle, and Felip Manyá. Transformations between Signed and Classical Clause Logic. *In In Proceedings, 29th International Symposium on Multiple-Valued Logics (ISMVL)*, pages 248-255. IEEE Press, 1999.

[4] Reiner Hahnle Bernhard Beckert and Felip Many´a. On the Regular 2-SAT Problem. *University of Karlsruhe, Dept. of Computer Science. Available at ftp://sonja.ira.uka.de/pub/beckert/Regular* 2SAT.ps.gz, 1999.

[5] M. Davis, G. Logemann, and D. Loveland. A Machine Program for Theorem Proving. *Comunnications of the ACM*, 5:394–397, 1962.

[6] W.F. Dowling and J.H. Gallier. Linear-time algorithms for testing the satisfiability of horn propositional formulae. *Journal of Logic Programming*, 3:267–284, 1984.

[7] Gonzalo Escalada-Imaz and Felip Manya. The Satisfiability Problem for Multiple-Valued Horn Formulae. *In Proceedings, International Symposium The SAT Problem of Signed CNF Formulas 21 on Multiple-Valued Logics(ISMVL)*, pages 250-256, 1994.

[8] Reiner Hahnle. Exploiting Data Dependencies Inmany-Valued Logics. *Journal of Applied Non-Classical Logics*, pages 49-69, 1996.

[9] REINERH¨DHNLE. Short Conjunctive Normal Forms in finitely Valued Logics. *Journal of Logic and Computation*, 1994.

[10] Felip Manyá. Proof Procedures Formultiple-Valued Propositional Logcs. MonografiesdeLSInstituddSInvertigacit'o en Inter.lig'encia Artificial.

[11] Felip Manyá. The 2-SAT Problem in Signd CNF Formulas. Multiple-Values Logic. *An International Journal*, 1999.

[12] A. Itai S. Even and A. Shamir. On the Complexity of Timetable and Multicommodity flow Problems. *SIAM Journal of Computing*, pages 691-703, 76.

[13] S.A.Cook. The Complexity of Theorem-Proving Procedures. *In In Proceedings of the Third ACM Symposium on theory of Computing*, pages 151-158, 1971.

[14] Viorica Sofronie-Stokkermans. On Translation of finitely-Valued Logics to Classical first-Order Logic. page pages 410-411, 1998.