Bibliografia

[1] http://archive.eiffel.com/doc/manuals/technology/contract/ariane/page.html

[2] http://cnews.canoe.ca/CNEWS/Space/2004/01/30/330400-ap.html

[3] http://www.informatik.uni-augsburg.de/swt/fmg/

[4] http://www-2.cs.cmu.edu/ modelcheck/smv.html

[5] M. PRESBURGUER. Über die Vollständigkeit enines gewissen Systems der Arithmetik ganzer Zahlen. Comptes-rendu du I congrès des mathématiciens de pays Slaves, Warsaw. 92–101,1929.

[6] G. KREISEL and H.WANG Some aplications of formalized consistency proofs. Fundamenta Mathematicae- 42,101–110,1955.

[7] R. FLOYD. Assigning meaning to programs. *Symposia in Applied Mathematics*, 19:19–32, 1967.

[8] G.KREISEL and A.LÉVY Reflection principles and their use for establishing the complexity of axiomatic systems Zeitschr. f. math.Logik und Grundlagen d. Math.- Bd. 14, 97–142,1968.

[9] C.A.R. HOARE. An axiomatic basis for computer programming. Communications of the ACM, 12(10): 576–580,583, 1969.

[10] C.A.R. HOARE, and WHIRTH, N. An axiomatic Definition of the Programming Language PASCAL. Acta Informatica, 2: 335–355, 1973. Springer-Verlag.

[11] PARIK, R.J. Some results on the length of proofs. *Transactions of the American Mathematical Society*, 177:29–36, March 1973.

[12] S. GOTO. Program synthesis from natural deductions proofs. *International Joint Conference on Artificial Intelligence-Tokyo*, 339–341, 1979.

[13] W.A. HOWARD. The Formulae-as-Types Notion of Construction". In Hindley J.R. and Seldin J.P., editors, *To H.B. Curry: Essays on combinatory logic, Lambda Calculus and Formalisation*, Academic Press, 1980.

[14] PAS. VELOSO. Outlines of a mathematical theory of general problems. *Philosophia Naturalis*, 21(2/4): 234–362, 1984.

[15] J.L. BATES and R.L. CONSTABLE. Proof as Programs. *ACM Transactions on Programming Languagens and Systems*, 7(1):113–136, 1985.

[16] S. YOCCOZ. Constructive aspects of the omega-rule: Application to proof systems in computer science and algorithmic logic. *LNCS 379*:553–565, 1989.

[17] A. BUNDY,A. SMAIL and G.A.WIGGINS. The synthesis of logic programs from inductive proofs. In J. Lloyd, editor, *Computational Logic*, 135–149, Springer-Verlag, 1990.

[18] S. BAKER, A.IRELAND and A.SMAILL. On the use of the constructive omega rule within automated deduction. *LNAI 624*:214-225.

[19] K. LAU and G. WIGGINS. A tutorial on Synthesis of Logic Programs form Specifications. In P. Van Hentenryck, editor, *Proceedings of the Eleventh International Conference on Logic Programming*, 11–14, MIT Press, 1994.

[20] G. J. HOLZMANN The Model Checker SPIN. In: Software Engineering 23 (5) pp. 279-295, 1997.

[21] J. CHAZARAIN, and S. MULLER. Automated synthesis of recursive programs from a ∀∃ logical Specification. *Journal of Automated Reasoning*, 21:233–275, 1998.

[22] C. KREITZ. Program synthesis - Automated Deduction - A basis for Applications. 105–134, Kluwer,1998.

[23] H. BENL, U. BEGER, H. SCHWICHTENBERG, M. SEISENBERGER,and W. ZUBER. Proof theory at work: Program development in the Minlog system. In W. Bibeland P.H.Schmitt, editors, *Automated Deduction* Kluwer, Vol II, 1998.

[24] E.H. HAEUSLER. Extracting Solution from Constructive Proofs: Towards a Programming Methodology. *Brasilian Eletronic Journal on Mathematics of Computation (BEJMC)*, No. 0- Vol 0,1999,(http://gmc.ucpel.tche.br/bejmc).

[25] U.BERGER, H.SCHWICHTENBERG e W.BUCHHOLZ. *Refined Program Extraction from Classical Proofs.* Annals of Pure and Applied Logic 114, 3–25, 2002.

[26] G.M.H. SILVA, E.H. HAEUSLER and P.A.S. VELOSO. Extracting Programs from Intuitionistic Proofs. In Abe, J.M and Filho, J.I.S., editors,*Advances in Logic, Artificial Intelligence and Robotics*, IOS Press and Ohmsha, Vol I, 2002.

[27] G.DOWEK, T.HARDIN e C. KIRCHNER. *Theorem proving modulo.* Journal of Automated Reasoning -31(1),33–72,2003.

[28] J.L. CALDWELL, P. IAN, J.G.UNDERWOOD. Search algorithms in Type Theory. *http://meru.cs.uwyo.edu/ jlc/papers.html*

[29] Truth definitions ans consistency proofs. Transactions of the American Mathematical Society -73,243–275,1952.

[30] J.R. SCHOENFIELD. *On a Restricted ω-rule.* Bull Acad. Sc. Polon, 7, 1959.

[31] D. PRAWITZ. *Natural Deduction - A Proof Theoretical Study.* Almavist and Wiksell, Stockolm, Goteborg-Uppsala,1965.

[32] KNUTH, D.E. *The art of computer programming - Fundamental algorithms.* Addison-Wesley, 1968.

[33] KNUTH, D.E. *The art of computer programming - Seminumerical algorithms.* Addison-Wesley, 1969.

[34] H.B. ENDERTON. *A Mathematical Introduction to Logic.* New York Academic Press, 1972.

[35] KNUTH, D.E. *The art of computer programming - Sorting and searching.* Addison-Wesley, 1973.

[36] C. CHANG and R.C. LEE. *Symbolic Logic and Mechanical Theorem Prover.* Academic Press,1973.

[37] E.G.K.LOPEZ-ESCOBAR. *On a extremely restricted ω-rule.* Fundamenta Mathematicae,90, 1976.

[38] DUMMETT, M. *Elements of intuitionism.* Oxford University Press, 1977.

[39] P. MARTIN-LÖF. *Intuitionistic Type Theory.* Edizioni di filosofia e Scienza, Biblioplolis, 1984.

[40] E.G.K.LÓPEZ-ESCOBAR e I.M.L.Ottaviano. *A regra ω: Passado, Presente e Futuro.* Coleção CLE, 1987.

[41] S. KLEENE. *An Introduction to Metamathematics.* North-Holland, 1988.

[42] J. GIRARD, Y. LAFONT and P.TAYLOR. *Proof and Types.* Cambrigde University Press,1989.

[43] C.B.JONES Systematic Software Sevelopment using VDM. C.A.R. Hoare Series Editor. Prentice Hall,1989.

[44] R.M. SMULLYAN *Gödels incompleteness theorems.* Oxford Logic Guides,19,1992.

[45] T.W. PRATT, M.V. ZELKOWITZ. *Programming Languages Design and Implementation*. Third Editon, Prentice Hall,1996.

[46] G.BOOLOS *The logic of provability*. Cambridge University Press, 1996.

[47] G.M.H. SILVA. *Um Estudo em Síntese Construtiva de Programas utilizando Lógica Intuicionista*. Dissertação de Mestrado - Departamento de Informática -PUC-Rio, 1999.

[48] STAA, A.V. *Programação modular : desenvolvendo programas complexos de forma organizada e segura*. Editora Campus, 2000.

[49] EPSTEIN R.L e CARNIELLI W.A. *Computability: computable functions, logic and the foundations of mathematics, with the timeline Computability and Undecidability*. Second edition. Wadsworth/Thomson Learning,Belmont, CA, 2000.

A
Apêndice

A.1
Prova de Correção

No capítulo 3, na prova de correção do processo de síntese construtiva (seção 3.4.1) é apresentada apenas a prova para as regras de inferência que tiveram os comandos da linguagem de programação associados a elas alterados, em relação a prova de correção apresentada em [47]. Esta seção apresenta a prova das outras regras que não foram tratadas anteriormente.

◇ Introdução da Conjunção

Seja esta regra a última a ser aplicada na derivação D:

$$D = \left\{ \begin{array}{c} \begin{array}{cc} \Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m & \Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \\ \vdots & \vdots \\ \Lambda : \alpha_T^V & \Psi : \beta_T^V \end{array} \\ \hline \Lambda \otimes \Psi : (\alpha \wedge \beta)_T^V \end{array} \right.$$

A regra de construção de programas, associada a essa regra de inferência realiza a composição dos programas associados as suas premissas.

Pela hipótese indutiva tem-se que:
1 - $\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Lambda : \alpha_T^V$ se, e somente se:
$\exists U_1$ completo $\subseteq CSC_M^{\theta}\left(\alpha_T^V\right)$ tal que, $\forall Q_1 \in U_1$ onde $Q = \left\langle L_1, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_1 \right\rangle$, tem-se:
$\underset{Hoare}{\vdash} \left\{ in = L_1 \wedge (\overrightarrow{v} = \overrightarrow{i}) \right\} \Lambda \{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W_1 \}$.

2 - $\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Psi : \beta_T^V$ se, e somente se:

$\exists U_2$ completo $\subseteq CSC_M^{\theta}\left(\beta_T^V\right)$ tal que, $\forall Q_2 \in U_2$ seja $Q = \left\langle L_2, \left\langle \overrightarrow{i}\ , \overrightarrow{o} \right\rangle, W_2 \right\rangle$, tem-se:

$$\underset{Hoare}{\vdash} \left\{ in = L_2 \wedge (\overrightarrow{v} = \overrightarrow{i}) \right\} \Psi \{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W_2 \} \ .$$

A composição dos comandos podem ser descritos pela seguinte semântica:

(a)

$$A = \left\{ in = L_1 \wedge (\overrightarrow{v} = \overrightarrow{b}) \right\} \Lambda \left\{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W_1 \right\}$$
$$B = \left\{ in = L_2 \wedge (\overrightarrow{v} = \overrightarrow{b}) \right\} \Psi \left\{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W_2 \right\}$$

$$\frac{A \quad B}{\left\{ in = (L_1 \wedge L_2) \wedge (\overrightarrow{v} = \overrightarrow{b}) \right\} \Lambda \otimes \Psi \left\{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = (W_1 \wedge W_2 \right\}}$$

Dado que:

$$CSC_M^{\theta}\left((\alpha \wedge \beta)_T^V\right) =$$

$$\left\{ \left\langle \widehat{L_1 L_2}, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, \widehat{W_1 W_2} \right\rangle \Big/ \begin{array}{c} k_\alpha = \left\langle L_1, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_1 \right\rangle \in CSC_M^{\theta}\left(\alpha_T^V\right) e \\ k_\beta = \left\langle L_2, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_2 \right\rangle \in CSC_M^{\theta}\left(\beta_T^V\right) \end{array} \right\}$$

Se $\left(U' \subseteq CSC_M^{\theta}\left((\alpha \wedge \beta)_T^V\right)\right.$ então:

$$U' = \left\{ \left\langle \widehat{L_1 L_2}, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, \widehat{W_1 W_2} \right\rangle \Big/ \begin{array}{c} \left\langle L_1, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_1 \right\rangle \in U_1 \ \ e \\ \left\langle L_2, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_2 \right\rangle \in U_2 \end{array} \right\}$$

Como U' é formado por U_1 e U_2, que possuem os mesmos valores de entrada e saída em memória$(\overrightarrow{i}, \overrightarrow{o})$, tem-se que U' é completo.

Seja $Q \in U'$, $Q = \left\langle \widehat{L_1 L_2}, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, \widehat{W_1 W_2} \right\rangle$ tal que: $Q_\alpha = \left\langle L_1, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_1 \right\rangle \in \left(U_1 \subseteq CSC_M^{\theta}\left(\alpha_T^V\right)\right)$ e $Q_\beta = \left\langle L_2, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_2 \right\rangle \in \left(U_2 \subseteq CSC_M^{\theta}\left(\beta_T^V\right)\right)$. Q_α e Q_β, pela hipótese indutiva, são gerados pelos programas Λ e Ψ respectivamente.

Pela semântica do comando gerado (a), tem-se que $(\Lambda \otimes \Psi)$ calcula Q, como Q é uma tupla arbitrária que pertence a U', pode-se concluir que:

$$\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Lambda \otimes \Psi : (\alpha \wedge \beta)_T^V$$

◇ Eliminação da Conjunção

Seja esta regra a última a ser aplicada:

Caso 1:

$$D = \left\{ \begin{array}{c} \Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \\ \vdots \\ \Lambda : (\alpha \wedge \beta)^V_T \\ \hline \Lambda : \alpha^V_T \end{array} \right.$$

Caso 2:

$$D = \left\{ \begin{array}{c} \Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \\ \vdots \\ \Lambda : (\alpha \wedge \beta)^V_T \\ \hline \Lambda : \beta^V_T \end{array} \right.$$

O programa associado à premissa calcula mais propriedades do que a conclusão faz referência. Entretanto para simplificar o processo de geração de programas, não serão utilizados filtros e o programa associado a conclusão será o mesmo da premissa.

Caso 1:

Pela hipótese indutiva:

$\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Lambda : (\alpha \wedge \beta)^V_T$ se, e somente se:

$\exists U$ completo $\subseteq CSC^{\theta}_M\left((\alpha \wedge \beta)^V_T\right)$ tal que, $\forall Q \in U$ sendo $Q = \left\langle L, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W \right\rangle$, tem-se que:

$\underset{Hoare}{\vdash} \left\{ in = L \wedge (\overrightarrow{v} = \overrightarrow{i}) \right\} \Lambda \{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W \}$.

Dado que:

$CSC^{\theta}_M\left((\alpha \wedge \beta)^V_T\right) =$

$$\left\{ \left\langle \widehat{L_1 L_2}, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, \widehat{W_1 W_2} \right\rangle \diagup \begin{array}{c} k_\alpha = \left\langle L_1, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_1 \right\rangle \in CSC_M^\theta \left(\alpha_T^V\right) e \\ k_\beta = \left\langle L_2, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_2 \right\rangle \in CSC_M^\theta \left(\beta_T^V\right) \end{array} \right\}$$

Pela definição de $CSC_M^\theta \left((\alpha \wedge \beta)_T^V\right)$, tem-se que $U_\alpha \subseteq U$, logo U_α é completo.

Seja $Q_\alpha \in \left(U_a \subseteq CSC_M^\theta \left(\alpha_T^V\right)\right)$ então $Q_\alpha = \left\langle L_1, \left\langle \overrightarrow{i}, \overrightarrow{o} \right\rangle, W_1 \right\rangle$.

Pode-se observar que $Q_\alpha \subseteq Q$. Como Q_α é uma tupla arbitrária de U_α, conclui-se que:

$$\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Lambda : \alpha_T^V$$

Caso 2: Essa prova é análoga à apresentada para o caso 1. Assim,

$$\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Psi : \beta_T^V$$

◇ Introdução da Disjunção

Seja esta a última regra aplicada em uma derivação:

Caso 1:

$$D = \left\{ \begin{array}{c} \Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \\ \vdots \\ \Lambda^{(i)} : \alpha_T^V \\ \hline \Lambda^{(i)} : (\alpha \vee \rho)_T^V \end{array} \right.$$

Caso 2:

$$D = \left\{ \begin{array}{c} \Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \\ \vdots \\ \Psi^{(j)} : \rho_T^V \\ \hline \Psi^{(j)} : (\alpha \vee \rho)_T^V \end{array} \right.$$

Esta regra pode ser interpretada da seguinte forma: existe uma propriedade que pode ser expressa de dois modos diferentes. Assim, tem-se dois programas diferentes que calculam a mesma propriedade, logo, qualquer um deles é suficiente para o cálculo da propriedade. Pode-se, então, assumir que a regra para a construção de programa associado a esta regra não altera o programa associado à premissa da mesma.

Caso 1:

Pela hipótese indutiva:

$\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Lambda^{(i)} : \alpha_T^V$ se, e somente se:

$\exists U$ completo $\subseteq CSC_M^\theta\left(\alpha_T^V\right)$ tal que, $\forall Q \in U$ onde $Q = \left\langle L_1, \left\langle \overrightarrow{i_1}, \overrightarrow{o_1} \right\rangle, W_1 \right\rangle$, tem-se que:

$\underset{Hoare}{\vdash} \left\{ in = L_1 \wedge (\overrightarrow{v} = \overrightarrow{i_1}) \right\} \Lambda^{(i)} \{ (\overrightarrow{t} = \overrightarrow{o_1}) \wedge out = W_1 \}$.

Pela hipótese indutiva: $U \subseteq CSC_M^\theta\left(\alpha_T^V\right)$, logo, $U \subseteq \left(CSC_M^\theta\left(\alpha_T^V\right) \cup CSC_M^\theta\left(\rho_T^V\right)\right)$.

Dado que:

$$CSC_M^{\theta}\left((\alpha \vee \rho)_T^V\right) = CSC_M^{\theta}\left(\alpha_T^V\right) \cup CSC_M^{\theta}\left(\rho_T^V\right)$$

Tem-se que: $U \subseteq CSC_M^{\theta}\left((\alpha \vee \rho)_T^V\right)$.

Logo, $\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Lambda^{(i)} : (\alpha \vee \rho)_T^V$

Caso 2: A prova para este caso é análoga à apresentada para o caso 1.

Assim: $\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Psi^{(j)} : (\alpha \vee \rho)_T^V$

◇ Eliminação da Disjunção

Suponha que esta regra seja a última aplicada em uma derivação D:

$$D = \left\{ \begin{array}{c} \begin{array}{ccc} \Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m & \sigma : \alpha_{T_1}^{V_1} & \sigma : \rho_{T_1}^{V_1} \\ \vdots & \vdots & \vdots \\ \sigma : (\alpha \vee \rho)_{T_1}^{V_1} & \Lambda : \gamma_T^V & \Psi : \gamma_T^V \end{array} \\ \hline if\ (\alpha)\ then(\Lambda)\ else\ (if\ (\rho)\ then(\Psi)) : \gamma_T^V \end{array} \right.$$

Na aplicação desta regra, o programa associado à conclusão é o comando condicional, que após a verificação de qual propriedade (descrita pelas premissas menores) é satisfeita, um conjunto de comandos será executado para obter a solução relacionada com a conclusão.

Pela hipótese indutiva:

1- $\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Lambda : \gamma_T^V$ se, e somente se:

$\exists U_1$ completo $\subseteq CSC_M^{\theta \cup \alpha}\left(\gamma_T^V\right)$ tal que, $\forall Q_1 \in U_1$ sendo $Q_1 = \left\langle L, \left\langle \overrightarrow{i}\ , \overrightarrow{o} \right\rangle, W \right\rangle$, tem-se que:

$\underset{Hoare}{\vdash} \left\{ in = L \wedge (\overrightarrow{v} = \overrightarrow{i}) \right\} \Lambda \{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W \}$.

2 - $\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} \Psi : \gamma_T^V$ se, e somente se:

$\exists U_2$ completo $\subseteq CSC_M^{\theta \cup \rho}\left(\gamma_T^V\right)$ tal que, $\forall Q_2 \in U_2$ sendo $Q_2 = \left\langle L, \left\langle \overrightarrow{i}\ , \overrightarrow{o} \right\rangle, W \right\rangle$, tem-se que:

$\underset{Hoare}{\vdash} \left\{ in = L \wedge (\overrightarrow{v} = \overrightarrow{i}) \right\} \Psi \{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W \}$.

Sendo

$A = \left\{ in = L \wedge (\overrightarrow{v} = \overrightarrow{i}) \wedge \alpha \right\} \Lambda \left\{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W \right\}$ e

$B = \left\{ in = L \wedge (\overrightarrow{v} = \overrightarrow{i}) \wedge \sim \alpha \wedge \rho \right\} \Psi \left\{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W \right\}$,

com base na semântica do comando condicional, a semântica do programa gerado será a seguinte:

$$\frac{A \quad B}{\left\{ in = L \wedge (\overrightarrow{v} = \overrightarrow{i}) \right\}\ if\ (\alpha)\ then\ \{\Lambda\}\ else\ \{if\ (\rho)\ then\ \{\Psi\}\} \left\{ (\overrightarrow{t} = \overrightarrow{o}) \wedge out = W \right\}}$$

Pela hipótese indutiva, a partir da prova γ com a hipótese α pode-se extrair o $CSC_M^{\theta \cup \alpha}\left(\gamma_T^V\right)$, e a partir da prova de γ com a hipótese ρ pode-se extrair o

$CSC_M^{\theta \cup \rho}\left(\gamma_T^V\right)$.

Dado que α e ρ são hipóteses exclusivas, tem-se que: $CSC_M^{\theta \cup \alpha}\left(\gamma_T^V\right) \cup CSC_M^{\theta \cup \rho}\left(\gamma_T^V\right) = CSC_M^{\theta}(\gamma)_T^V$.

Se $U' = CSC_M^{\theta}(\gamma)_T^V$, então $U' = U_1 \cup U_2$. Como U' é construído a partir de U_1 e U_2, U' é completo.

Logo:

$$\Delta, \beta_1, ..., \beta_n, \delta_1, ..., \delta_m \underset{M,\sigma}{\models} if\ (\alpha)\ then(\Lambda)\ else\ (if\ (\rho)\ then(\Psi)) : \gamma_T^V$$

Observação A.1.1 *A premissa maior desta regra pode ter conteúdo computacional, mas o programa associado, após a sua execução, retornará valores boolenos "Verdadeiro"ou "Falso". Logo, podem ser interpretados como tendo apenas conteúdo computacional.*

A.2
Prova do teorema apresentado por Parik em [11]

Teorema A.1 *Parik:*
$\underset{PA}{\vdash} \forall x \alpha(x)$ *se, e somente se, existe um* k *tal que* $\forall n \overset{k}{\underset{PA}{\vdash}} \alpha(n)$ *[11]*

Será apresentado um esquema da prova deste teorema, pois é necessário fazer uma análise desta prova para provar o teorema 4.5.

Observação A.2.1 *Note que o resultado do teorema 4.9 é utilizado sobre HA. Logo, a prova será restringida sobre os operadores lógicos intuicionistas.*

A.2.1
Conceitos Básicos

✠ Notação

Abaixo segue as notações utilizadas na formalização do sistema:

1. Cálculo de predicados
   (a) Variáveis : $x, y, z, x_1, ...$
   (b) Constantes: $c, c_1, c_{11}, ...$
   (c) Símbolos de predicados n-ários para $n \geqq 0$: $F, G(x), H(x, y), ...$
   (d) Símbolos funcionais: $f, g, h, f_1, ...$
   (e) Símbolos lógicos: $\neg$, $\vee$, $\wedge$, $\rightarrow$, $\forall$, $\exists$.
   (f) [,].
2. Meta notação
   (a) Metavariáveis: u,v,w,u_1,...,
   (b) Variáveis de termos: r,s,t,r_1,...,
   (c) Variáveis para predicados n-ários (fórmulas) com n$\geqq$0: $P, Q(x), R(x, y), ...$

Observação A.2.2 *Pode-se assumir, sem perda de generalidade, que os* $n's$ *nos itens (c) são limitados, apesar de se saber que são necessários infinitos predicados de variáveis n-ários para cada n.*

Os termos são formados a partir de variáveis, constantes, metavariáveis, variáveis de termo e através de símbolos funcionais. Letras gregas minúsculas denotam termos. Se estes não possuirem meta notação serão denominados como termos regulares e podem ser denotados pelas iniciais do alfabeto grego (α,β,...).

As fórmulas atômicas são formadas por variáveis de predicados ou constantes adição de termos, caso contrário serão formadas por fórmulas atômicas combinadas com conectivos e quantificadores $\forall x$, $\exists x$,$\forall u$, $\exists u$,.... Se as fórmulas forem regulares, estas serão denotadas por letras maiúsculas do alfabeto romano $(A, B...)$, caso contrário serão representadas por letras góticas $(\mathfrak{F}, \mathfrak{G})$.

✠ Processo de substituição

Uma substituição $\mathcal{S}$ será a atribuição de:

- variáveis $x_1, ..., x_n$ para determinadas metavariáveis $u_1, ..., u_n$,

- termos regulares $\alpha_1, ...\alpha_m$ para determinados termos de variáveis $t_1, ..., t_m$, e

- fórmulas regulares $A, B, ...$ para determinadas variáveis de predicado $P, Q(x), ...$.

O mapeamento das substituições de termos regulares é definido por:

$\mathcal{S}(u_i) = x_i, \mathcal{S}(t_i) = \alpha_i \quad i = 1, ..., n, j = 1, ...m$, e

$\mathcal{S}(f(\sigma_1, ..., \sigma_k)) = f(\mathcal{S}(\sigma_1), ..., \mathcal{S}(\sigma_k))$

onde a expressão do lado direito da atribuição é definida e f é um símbolo de função k-ário.

Para uma dada fórmula $\mathfrak{F}$, a substituição $\mathcal{S}(\mathfrak{F})$ é definida por:

Definição A.2 *(i)Seja $\mathfrak{F}$ atômico com o símbolo de predicado regular onde $\mathfrak{F} = \mathcal{F}(\sigma_1, ..., \sigma_k)$. Então $\mathcal{S}(\mathfrak{F}) = \mathcal{F}(\mathcal{S}(\sigma_1), ..., \mathcal{S}(\sigma_k))$, dado que a substituição da expressão do lado direito da igualdade seja definida.*
(ii)Seja $\mathfrak{F}$ atômico para um símbolo de predicados de variável $P(x_1, ..., x_k)$ como símbolo de predicado: $\mathfrak{F} = P(x_1, ..., x_k), \mathcal{S}(P(x_1, ..., x_k)) = A$. Então $\mathcal{S}(\mathfrak{F}) = s(x_1, ...x_k; \mathcal{S}(\sigma_1), ..., \mathcal{S}(\sigma_k))A$, onde s é a substituição usual e apenas as ocorrências de x_i livres serão substituídas.
(iii)$\mathcal{S}(\mathfrak{F} \wedge \mathfrak{G}) = \mathcal{S}(\mathfrak{F}) \wedge \mathcal{S}(\mathfrak{G})$ etc.,
(iv)$\mathcal{S}(\forall x\mathfrak{F}) = \forall x\mathcal{S}(\mathfrak{F})$,
(v)$\mathcal{S}(\forall u\mathfrak{F}) = (\forall \mathcal{S}(u))\mathcal{S}(\mathfrak{F})$.

✠ Restrições admissíveis

Uma restrição é dita admissível se for da forma:
- "σ é livre para u (o mesmo vale para x) em P"ou
- "u (ou x) não é livre em P"ou
- "u (ou x) não ocorre em P",

onde u é uma metavariável, x é uma variável, σ é um termo e P é uma variável de predicado.
Exemplo de uma substituição $\mathcal{S}$ que obedece a restrição $R : \mathcal{S}$ obedece "σ é livre para u em P"se , e somente se, $\mathcal{S}(\sigma)$ é livre para $\mathcal{S}(u)$ em $\mathcal{S}(P)$. Uma

substituição obedece um conjunto finito de restrições se, e somente se, todas elas forem obedecidas. Utiliza-se a expressão *restrição* ao invés de *restrições admissíveis*.

Desta forma, um esquema de axiomas será representado por um par ordenado $(\mathfrak{F}, R)$, onde $\mathfrak{F}$ é uma fórmula e R é um conjunto finito de restrições. Axiomas que satisfazem um esquema de axiomas serão fórmulas $\mathcal{S}(\mathfrak{F})$, onde $\mathcal{S}$ obecede R. Similarmente, uma regra de inferência $k-$ária é uma seqüência $(\mathfrak{F}_1, ..., \mathfrak{F}_k, \mathfrak{G}, R)$, $\mathfrak{F}_1, ..., \mathfrak{F}_k$, onde $\mathcal{G}$ são fórmulas e R é um conjunto finito de restrições, e esta regra será aplicada da seguinte forma: "Se $A_1, ..., A_k, B$ são $\mathcal{S}(\mathfrak{F}_1), .., \mathcal{S}(\mathfrak{F}_k), \mathcal{S}(\mathcal{G})$, respectivamente $\mathcal{S}$ obedece R, então B é derivado a partir de $A_1, ..., A_k$ ".

Neste trabalho, teorias são formalizadas com um número finito de axiomas, e os sistemas são definidos por um número finito de esquemas de axiomas e de esquemas de regras de inferência.

A.2.2

Lemas de substituição

Dado uma prova em um sistema de esquemas, a qual pode ser estruturada na forma de uma árvore de prova, uma análise desta prova é um conjunto de informações sobre a árvore de forma que, para cada fórmula, tem-se se é um axioma ou não, se for, a qual esquema pertence e, se for derivado, de qual regra de inferência é derivada.

Através de uma análise, deseja-se encontrar uma árvore de tais observações que não precise estar associada com alguma prova especificamente. Como existe apenas um número finito de análises de tamanho k, questões considerando provas de tamanho k podem ser reduzidas a questões sobre provas com uma análise particular $\mathfrak{A}$. O seguinte lema permite que se trabalhe com a noção de um objeto sintático ao invés da noção informal de análises.

O seguinte lema prova que toda análise $\mathfrak{A}'$ se reflete em uma árvore de prova. Essa análise pode ser "concatenada"com outras análises, de forma a obter uma análise $\mathfrak{A}$ que, após o processo de substituição $\mathcal{S}$ refletirá a prova da fórmula A e vice-versa.

Lema A.2.3 *Dado uma análise $\mathfrak{A}$, pode-se efetivamente encontrar fórmulas $\mathfrak{F}_1, ..., \mathfrak{F}_m$;$\mathcal{G}_1$,...,$\mathcal{G}_m$;$\mathcal{H}$ com um conjunto finito de restrições R, tal que a fórmula A tenha uma prova cuja análise é $\mathfrak{A}$ se, e somente se, existir uma substituição $\mathcal{S}$*

obedecendo R, tal que, $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_i), i = 1, ..., m$ *e* $\mathcal{S}(\mathcal{H}) = A$.
E a sequência $\mathcal{S}(\mathfrak{F}_1), ..., \mathcal{S}(\mathfrak{F}_m), A$ *é a prova desejada.*

Prova. A prova é realizada por indução no tamanho da análise de $\mathfrak{A}$ (k).

a) Se $k = 1$, então a análise pode ser reduzida para um esquema de axioma, logo $m = 0$;

b) $k > 1$. Seja o último passo de $\mathfrak{A}$ a aplicação da regra $(\mathfrak{L}_1, ..., \mathfrak{L}_l, \mathfrak{M}, R_l)$ sobre uma determinada fórmula previamente derivada.

Suponha que a análise da fórmula previamente derivada seja a seqüência $\mathfrak{F}_1^1, ..., \mathfrak{F}_m^1 ; \mathcal{G}_1^1, ..., \mathcal{G}_m^1; \mathcal{H}_1; ..., \mathfrak{F}_1^l, ..., \mathfrak{F}_m^l; \mathcal{G}_1^l, ..., \mathcal{G}_m^l; \mathcal{H}_l$.

Pode-se assumir que todos os meta-símbolos desses diferentes grupos tenham sido renomeados, de forma a não existir metavariáveis comuns entre eles ou com $\mathfrak{L}_1, ..., \mathfrak{L}_l, \mathfrak{M}$.

A sequência para a análise $\mathfrak{A}$ será $\mathfrak{F}_1^1 ..., \mathfrak{F}_m^1, \mathcal{H}_1, \mathfrak{F}_1^2, ..., \mathcal{H}_l; \mathcal{G}_1^l, ..., \mathcal{G}_m^l, \mathfrak{L}_1, \mathcal{G}_1^2, ..., \mathfrak{L}_l; \mathfrak{M}$. As restrições são obtidas pela união de todas as diferentes restrições.

Suponha que exista uma substituição $\mathcal{S}$ para a sequência acima, i.e.: $\mathcal{S}$ $(\mathfrak{F}_1^1) = \mathcal{S}$ $(\mathcal{G}_1^l), \mathcal{S}$ $(\mathcal{H}_1) = \mathcal{S}$ $(\mathfrak{L}_1)$ e $\mathcal{S}$ $(\mathfrak{M}) = A$.

Então, pela hipótese indutiva, cada $\mathcal{S}$ $(\mathcal{H}_1)$ é provado pela subprova correspondente e $\mathcal{S}$ $(\mathfrak{M}) = A$ será provado a partir de $\mathcal{S}$ $(\mathfrak{L}_j)$ utilizando a regra de inferência associada. Porém, se $\mathcal{S}$ $(\mathcal{H}_i) = \mathcal{S}$ $(\mathfrak{L}_i)$, tem-se a prova desejada.

Para provar o outro sentido da implicação, suponha que se tenha a prova desejada. Desta forma, cada premissa da última regra poderá ser provada a partir da subprova correspondente, e a última fórmula A é provada com a última regra de inferência. Logo, existirá um processo de substituições $\mathcal{S}_1, ..., \mathcal{S}_l, \mathcal{S}'$ para todas as regras. Como todas as metavariáveis foram renomeadas de forma a evitar conflitos, pode-se combinar todas as substituições gerando a substituição $\mathcal{S}$. □

Notação

PA símboliza o sistema usual da aritmética de Peano com S, +, . e o esquema da indução finita, PA^* corresponde o sistema de PA, onde as operações +,. são substituídas pelos predicados ternários A, M e com a adição de axiomas que dizem que esses predicados representam funções. Se T for um dos dois sistemas, então T_l é um sistema com a indução restrita a fórmulas lógicas de complexidade $\leqq l$. A complexidade lógica é dada pelo número de símbolos lógicos

em uma fórmula. Uma teoria $T, \vdash_l^k A$ representa que A é provavel em T com no máximo k aplicações de regras de inferência. PA, PA^* possuem o mesmo poder de expressão, mas, provas em PA^* serão, geralmente, mais longas. A aritmética de Presburger expressa uma aritmética com S,+,0, no entanto, sem a multiplicação. É sabido por [5] que PA (ou PA^*) é completo com respeito a fórmulas fechadas na qual a multiplicação não aparece.

Este lema prova que, dado um conjunto de fórmulas, suas restrições (isto é, uma análise) e uma fórmula α na aritmética de Presburger, de forma que existe um processo de substituições que obedece as restrições e quando aplicadas sobre o conjunto de fórmulas gera uma prova para α. Inicialmente serão realizadas todas as substituições de forma a se obter apenas fórmulas regulares em PA^*. No passo seguinte é demonstrado que as substituições são finitas e definem uma determinada fórmula na aritmética de Presburger.

Lema A.2.4 *Seja* $\mathfrak{F}_1$,...,$\mathfrak{F}_m$,$\mathcal{G}_1$,...,$\mathcal{G}_l$,$A_1(x)$,...,$A_p(x)$, *R tal que* $\mathfrak{F}_1$,...,$\mathfrak{F}_m$,$\mathcal{G}_1$,...,$\mathcal{G}_l$ *são fórmulas de* PA^*, $A_1(x)$,...,$A_p(x)$ *são fórmulas regulares de* PA^*, $m = l + p$, *e R é o conjunto de restrições. Então, existe uma fórmula* $B(x)$ *da aritmética de Presburger tal que, para todo n,* $B(n)$ *é equivalente a proposição: "Existe uma substituição* $\mathcal{S}$ *que obedece R tal que* $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_i)$ *para* $i \leqq l$ *e* $\mathcal{S}(\mathfrak{F}_{l+i}) = A_i(n)$ *para* $i \leqq p$*".*

Prova. A primeira parte do argumento é aplicado para todos os sistemas de esquemas com $0, S$ como símbolos. Os últimos passos são aplicados somente a PA^*. Fixando n e denotando $A_i(n)$ pela expressão $\mathcal{G}_{l+i}$. Então, precisa-se encontrar um $\mathcal{S}$ obedecendo R tal que $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_i)$ é uma fórmula regular de PA^* para $i \leqq m$.

Observação A.2.5 *Seja* $u = \Pi 3^{c_i}$ *onde* c_i *é o número de símbolos lógicos em* $\mathcal{G}_i$. *Se* $u = 1$, *então todas as fórmulas* $\mathcal{G}_i$ *são atômicas. Considere as relações minimais sobre as variáveis de predicados:* $P \leqslant Q, P \sim Q$ *de tal forma que* $\leqslant$ *é uma relação transitiva e* $\sim$ *é uma relação de equivalência e* $(P \leqslant Q) \wedge (Q \leqslant P) \leftrightarrow P \sim Q$. *Se P ocorrer em* $\mathfrak{F}_i$ *e Q ocorrer em* $\mathcal{G}_i$ *então* $Q \leqslant P$*; e se* $\mathcal{G}_i$ *for atômico então* $Q \sim P$. *Este sistema é decidível se tal relação existir e caso contrário não existe uma substituição* $\mathcal{S}$. *Por outro lado, pode-se ter* $P \leqslant Q$ *quando o número de símbolos lógicos em* $\mathcal{S}(P)$ *é menor ou igual ao número de símbolos lógicos em* $\mathcal{S}(Q)$.

Caso A.2.6 *Suponha* $u = 1$. *Neste caso* $\mathfrak{F}_i$ *e* $\mathcal{G}_i$ *são atômicos. Se* $x_1, ..., x_n$ *aborda todas as variáveis e* $u_1, ..., u_m$ *todas as metavariáveis que ocorrem em* $\mathfrak{F}_i, \mathcal{G}_i, R$. *Então pode-se assumir sem perda de generalidade que todos os* $\mathcal{S}(u_j)$ *são representados por* $x_1, ..., x_{n'}, ..., x_{n'+m}$. *Elimina-se, desta forma, todas as metavariáveis.*

Caso A.2.7 *Suponha $u > 1$, então existe um P tal que P é maximal em $\leqslant$, P ocorre em $\mathfrak{F}_i$, e $\mathcal{G}_i$ não é atômica.*

A prova é realizada analisando sempre o conectivo principal da fórmula $\mathcal{G}$.

Símbolo lógico: conjunção

Seja $\mathcal{G}_i$ da forma $\mathcal{G}_i^{'} \wedge \mathcal{G}_i^{''}$, substituindo P por $P' \wedge P''$, onde P' e P'' são duas novas variáveis de predicados, substitua o par de fórmulas que está sendo analisado, $\mathfrak{F}_i = P(\_)$, $\mathcal{G}_i = \mathcal{G}_i^{'}(\_) \wedge \mathcal{G}_i^{''}(\_)$, pelos pares $P'(\_)$, $\mathcal{G}_i^{'}(\_)$ e P'', $\mathcal{G}_i^{''}(\_)$. As restrições sobre P são refletidas em termos das restrições sobre P', P'', etc.

A substituição existirá se, dado que $\mathfrak{F}_i = P' \wedge P''$ e $\mathcal{G}_i$ da forma $\mathcal{G}_i^{'} \wedge \mathcal{G}_i^{''}$ existe um $\mathcal{S}$ tal que: $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_1)$ se, e somente se, $\mathcal{S}(P') = \mathcal{S}(\mathcal{G}_i^{'})$ e $\mathcal{S}(P'') = \mathcal{S}(\mathcal{G}_i^{''})$. Desta, forma obtem-se a correspondência entre as subfórmulas de $\mathfrak{F}_i$,e $\mathcal{G}_i$.

Símbolo lógico: disjunção

Seja $\mathcal{G}_i$ da forma $\mathcal{G}_i^{'} \vee \mathcal{G}_i^{''}$, substituindo P por $P' \vee P''$, onde P' e P'' são duas novas variáveis de predicados, substitua o par que está sendo analisado, $\mathfrak{F}_i = P(\_)$, $\mathcal{G}_i = \mathcal{G}_i^{'}(\_) \vee \mathcal{G}_i^{''}(\_)$, pelos pares $P'(\_)$, $\mathcal{G}_i^{'}(\_)$ e P'', $\mathcal{G}_i^{''}(\_)$. As restrições sobre P são refletidas em termos das restrições sobre P', P'', etc.

A substituição existirá se, dado que $\mathfrak{F}_i = P' \vee P''$ e $\mathcal{G}_i = \mathcal{G}_i^{'} \vee \mathcal{G}_i^{''}$ existe um $\mathcal{S}$ tal que: $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_1)$ se, e somente se, $\mathcal{S}(P') = \mathcal{S}(\mathcal{G}_i^{'})$ e $\mathcal{S}(P'') = \mathcal{S}(\mathcal{G}_i^{''})$. Desta forma, obtem-se a correspondência entre as subfórmulas de $\mathfrak{F}_i$,e $\mathcal{G}_i$.

Símbolo lógico: implicação

Seja $\mathcal{G}_i$ da forma $\mathcal{G}_i^{'} \rightarrow \mathcal{G}_i^{''}$, substituindo P por $P' \rightarrow P''$, onde P' e P'' são duas novas variáveis de predicados, substitua o par que está sendo analisado, $\mathfrak{F}_i = P(\_)$, $\mathcal{G}_i = \mathcal{G}_i^{'}(\_) \rightarrow \mathcal{G}_i^{''}(\_)$, pelos pares $P'(\_)$, $\mathcal{G}_i^{'}(\_)$ e P'', $\mathcal{G}_i^{''}(\_)$. As restrições sobre P são refletidas em termos das restrições sobre P', P'', etc.

A substituição existirá se, dado que $\mathfrak{F}_i = P' \rightarrow P''$ e $\mathcal{G}_i = \mathcal{G}_i^{'} \rightarrow \mathcal{G}_i^{''}$ existe um $\mathcal{S}$ tal que: $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_1)$ se, e somente se, $\mathcal{S}(P') = \mathcal{S}(\mathcal{G}_i^{'})$ e $\mathcal{S}(P'') = \mathcal{S}(\mathcal{G}_i^{''})$. Desta forma, obtém-se a correspondência entre as subfórmulas de $\mathfrak{F}_i$ e $\mathcal{G}_i$.

Símbolo lógico: negação

Seja $\mathcal{G}_i$ da forma $\neg\mathcal{G}_i^{'}$, substituindo P por $\neg P'$, onde P' é uma nova variável de predicado, substitua o par que está sendo analisado, $\mathfrak{F}_i = P(\_)$, $\mathcal{G}_i = \neg\mathcal{G}_i^{'}$, pelo par $P'(\_)$, $\mathcal{G}_i^{'}(\_)$. As restrições sobre P são refletidas em termos das restrições sobre P', etc.

A substituição existirá se, dado que $\mathfrak{F}_i = \neg P'$e $\mathcal{G}_i$= $\neg\mathcal{G}_i^{'}$ existe um $\mathcal{S}$ tal que: $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_1)$ se, e somente se, $\mathcal{S}(P') = \mathcal{S}(\mathcal{G}_i^{'})$. Desta forma, obtem-se a

correspondência entre as subfórmulas de $\mathfrak{F}_i$,e $\mathcal{G}_i$.

Símbolo lógico: Quantificador Universal

Seja $\mathcal{G}_i$ da forma $\forall x\mathcal{G}_i^{'}$, substituindo P por $\forall xP'$, onde P' é uma nova variável de predicado, substitua o par que está sendo analisado, $\mathfrak{F}_i = P(\_)$, $\mathcal{G}_i = \forall x\mathcal{G}_i^{'}$, pelo par $P'(\_)$, $\mathcal{G}_i^{'}(\_)$. A restrição "σ é livre para y em P"será substituída pela "y não é livre em P"ou pela "x não ocorre em σ"se $x \neq y$.

A substituição existirá se, dado que $\mathfrak{F}_i = \forall xP'$ e $\mathcal{G}_i$=$\forall x\mathcal{G}_i^{'}$ existe um $\mathcal{S}$ tal que: $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_1)$ se, e somente se, $\mathcal{S}(P') = \mathcal{S}(\mathcal{G}_i^{'})$. Desta forma, obtém-se a correspondência entre as subfórmulas de $\mathfrak{F}_i$e $\mathcal{G}_i$.

Símbolo lógico: Quantificador Existencial

Seja $\mathcal{G}_i$ da forma $\exists x\mathcal{G}_i^{'}$, substituindo P por $\exists xP'$, onde P' é uma nova variável de predicado, substitua o par que está sendo analisado, $\mathfrak{F}_i = P(\_)$, $\mathcal{G}_i = \exists x\mathcal{G}_i^{'}$, pelo par $P'(\_)$, $\mathcal{G}_i^{'}(\_)$. Se as variáveis livres do termo que substitui a variável x tiverem a restrição "σ é livre para y em P" estas serão substituídas pela "y não é livre em P". Após todas essas substituições será inserida a restrição "x não ocorre em σ".

A substituição existirá se, dado que $\mathfrak{F}_i = \exists xP'$ e $\mathcal{G}_i$= $\exists x\mathcal{G}_i'$ existe um $\mathcal{S}$ tal que: $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_1)$ se, e somente se, $\mathcal{S}(P') = \mathcal{S}(\mathcal{G}_i^{'})$. Desta forma, obtem-se a correspondência entre as subfórmulas de $\mathfrak{F}_i$e $\mathcal{G}_i$.

Esse procedimento deve ser aplicado a todos os símbolos de predicados equivalentes a P, e desta forma, a complexidade de u decresce. Depois de um número finito de passos, tem-se que $u = 1$, e todos os $\mathcal{G}_i$ serão atômicos. (Se não for possível executar o procedimento, então, não existe $\mathcal{S}$).

Assumindo, sem perda de generalidade, que $\mathcal{S}(\mathfrak{F}_i)$ e $\mathcal{S}(\mathcal{G}_i)$ são atômicos.

Sejam $x_1, ..., x_n$ todas as possíveis variáveis que podem ocorrer R. Se existir uma substituição $\mathcal{S}$, pode-se definir $\mathcal{S}'(P)$ como sendo a subfórmula atômica mais a esquerda de $\mathcal{S}(P)$ precedidos por todos quantificadores $\forall x_i$, $\exists x_i$ com $i = 1..n$ do seu específico escopo. Logo, se $\mathcal{S}$ for uma solução, $\mathcal{S}'$ também será.

Pode-se assumir que são conhecidos todos os símbolos de predicados envolvidos. Para cada par $\mathfrak{F}_i = P(\_), \mathcal{G}_i = F(\_)$, o símbolo de predicado para a variável de predicado P é F.

Todas as variáveis de predicado não cobertas por essa consideração, pode se atribuir a elas uma fórmula regular fixa A sem perda de generalidade.

Assim, pode-se assumir que, para cada variável de predicado P:
$\mathcal{S}\ (\mathcal{P}(y_1, ..., y_k)) = (Q_1 x_1), ..., (Q_p x_p) F_p(\theta_1(y_1, ..., y_k), ..., \theta_q(y_1, ..., y_k))$
onde Q_i são quantificadores, F_p é um predicado conhecido e θ_i são termos que dependem de y_i.

Pode-se dizer que duas fórmulas (não obviamente equivalentes) são iguais se, e somente se, os termos correspondentes forem iguais.

A argumentação dada até este momento se restringe a esquemas que conténham apenas S e 0. Neste ponto, a argumentação será extendida de forma a atender esquemas em PA^*.

Dado que α seja uma fórmula atômica em PA^*, α é da seguinte forma:

i - $\theta = \theta'$,

ii -$A(\theta, \theta', \theta'')$ ou $M(\theta, \theta', \theta'')$

onde θ, θ' e θ'' são da forma $S^q 0$ ou $S^q x$ e x é uma variável que está limitada a um conjunto finito de possibilidades.

Assim a substituição pode ser definida completamente pela especificação se (a) para cada F_p, se este for os predicados $=$, A ou M; ou se cada θ representa um $S^q 0$ ou $S^q x$ e, (b) especificando os valores q_i (metavariáveis), onde θ_i é igual a $S^{q_i} 0$ ou $S^{q_i} x$ ou uma variável de termo t_i igual a $S^{q_i} 0$; ou $S^{q_i} x$.

A parte (a) consiste em escolher uma das possibilidade dentre um conjunto finito. Para cada possibilidade, a condição sobre a substituição $\mathcal{S}$ para que $\mathcal{S}(\mathfrak{F}_i) = \mathcal{S}(\mathcal{G}_i)$ reduz para os vários q_i, satisfazendo determinadas equações da aritmética de Presburger. Assim, ao considerar que todas as possíveis substituições $\mathcal{S}$ para cada n em particular, obtém-se a seguinte equivalência: $\mathcal{S}$ existe se, e somente se, a fórmula $B_1(n) \vee ... \vee B_r(n)$ for válida, onde B_i é uma fórmula da aritmética de Presburger. Tomando $B(x)$ como sendo $B_1(n) \vee ... \vee B_r(n)$, temos prova do A.2.4. □

O teorema abaixo prova que, a partir do tamanho da árvore de prova e da complexidade máxima das fórmulas de uma árvore de prova, é possível calcular a complexidade máxima das fórmulas nas quais se pode aplicar a regra de inferência da indução.

Teorema A.3 *Seja T uma teoria em PA ou PA^* e $k, k' \in N$. Existe um número l calculado a partir de k, k' tal que, para cada fórmula A de complexidade $\leqq k'$, $\vdash^k_T A$ se, e somente se, $\vdash^k_{T_l} A$.*

Prova. Seja uma análise $\mathfrak{A}$ de k linhas e as substituições para as sequências $\langle \mathfrak{F}_1, ..., \mathfrak{F}_m, \mathcal{H}, ; \mathcal{G}_1, ..., \mathcal{G}_m, A, R \rangle$, onde $\mathfrak{F}_1, ..., \mathfrak{F}_m, \mathcal{G}_1, ...\mathcal{G}_m, \mathcal{H}$ é a sequência fornecida pelo A.2.3.

Seja $c(A) \leq k'$, então existe um número finito de maneiras nas quais A pode ser decomposta em uma fórmula atômica - a prova deste argumento é similar à prova da existência de um número finito de substituições apresentado no lema A.2.4.

Suponha que a complexidade máxima de uma fórmula que ocorre como axioma, esquema de axioma ou esquema de regra de inferência seja α. Então a complexidade de qualquer $\mathfrak{F}_i$,$\mathcal{G}_i$ ou $\mathcal{H}$ será no máximo α. Seja $m = k - 1$, então o valor inicial de u será $u_0 = 3^{\alpha(2k-1)}3^{k'1}$ (Supõe-se que os ramos de prova utilizados para a obtenção das fórmulas de m possuem a complexidade máxima α e, dado que no caso máximo o conectivo lógico aplicado pela regra de inferência é binário, observa-se que a complexidade das premissas de m é $\alpha(2k-1)$, e a complexidade da fórmula corrente é k').

Ao observar o lema A.2.4 tem-se que o valor de u será reduzido até atingir uma fórmula atômica ($u = 1$). Se a complexidade máxima das fórmulas sobre as quais pode-se aplicar a indução $m(u)$ de uma fórmula em determinado ponto desta redução for x, então a complexidade máxima no ponto anterior (premissas) não pode ser maior que $2x + 1$. Sendo $m(1) = k$ quando as fórmulas forem atômicas, e $m(u + 1) \leq 2m(u) + 1$; pode-se perceber que l é igual a $m(u_0) \leq 3^{u_0}$. □

Teorema A.4 *$\vdash_{PA^*} (\forall x)A(x)$ se, e somente se, existe um k tal que $(\forall n) \vdash^k_{PA^*} A(n)$.*

Prova. Se $\vdash_{PA^*} (\forall x)A(x)$ então existe um k tal que $(\forall n) \vdash^k_{PA^*} A(n)$, a prova de $(\forall x)A(x)$ de l linhas fornece provas uniformes de $A(n)$ para todo n com $l + 2$ linhas.

Se existe um k tal que $(\forall n) \vdash^k_{PA^*} A(n)$ então $\vdash_{PA^*} (\forall x)A(x)$.

Suponha que $(\forall n) \vdash^k_{PA^*} A(n)$, dado que $\vdash^k_{PA^*} A(n)$ correspondente a fórmula $B(x)$ na aritmética de Presburger tem-se que $(\forall x)B(x)$ é verdadeira, logo, $\vdash_{PA^*} (\forall x)B(x)$. Pelos lemas A.2.3 e A.2.4, existe uma fórmula $B_i(x)$ na aritmética de Presburguer tal que:
$\vdash^k_{PA^*} B(x)$ se, e somente se, $B_i(x)$, onde $B(x) = B_1(x) \vee ... \vee B_r(x)$, $i = 1...r$ e k é o tamanho da árvore de prova relacionada com $B(x)$. Como a aritmética de Presburguer é decidível, é possível saber qual é o $B_i(x)$ verdadeiro para cada

[1] A fórmula que calcula a complexidade de uma fórmula é: $\Pi 3^{ci}$.

n e, assim, pode-se calcular o tamanho de sua árvore de prova (k). Desta forma $\vdash_{PA^*} (\forall x)B(x)$ *pode ser representada como* $\vdash_{PA^*} (\forall n) \vdash^k_{PA^*} A(n)$. *Pelo teorema A.3:* $\vdash_{PA^*} (\forall n) \vdash^k_{PA^*_l} A(n)$. *A partir deste momento o tamanho de k não é mais necessário, logo ele pode ser omitido.*
Pela prova do lema A.3.1 tem-se que uma definição de verdade para PA^*_l *pode ser dada através de* PA^*.

Logo: $\vdash_{PA^*} (\forall n) \vdash_{PA^*} A(n)$. □

A.3
Prova do lema Kreisel e Wang em [6]

Lema A.3.1 *Para cada inteiro n, $Con(Z^n)$ pode ser provada em Z.*

Será apresentado um esquema da prova deste teorema, pois é necessário fazer uma análise desta prova para provar os teoremas 4.9 e 4.5.

Observação A.3.2 *Como o teorema 4.5 trabalha com o sistema HA e na prova do teorema 4.9 foi adotada a restrição de trabalhar apenas com os operadores lógicos intuicionistas, esta mesma restrição também será mantida aqui.*

Definição A.5 *$H_1^m(n), ..., H_m^m(n)$ são funções de substituição em m.*

Definição A.6 *Considerando que as variáveis das fórmulas são dadas por v_1, v_2, v_3, etc, $D(m, a, n)$ é número da fórmula obtida a partir da fórmula A (cujo número é a) resultante da substituição da variável v_i, $i \leqq m$ por $H_i^m(n)$ e v_j, $j > m$ por 0. Não é necessário que todos os v_i ocorram em A. Se A for uma fórmula fechada, i.e., v_i são variáveis livres, então: $D(m, a, n, x) = D(0, a, 0, a)$*

Definição A.7 *$P(a, b)$ se, e somente se, a é o número que representa a prova de b.*

Definição A.8 *Seja a e b números que representam as fórmulas A e B respectivamente, e sejam elas conectadas através de operadores lógicos binários, então $t(a, b)$ é o número de $A|B$, onde o operador $|$ pode ser substituído pelos operadores lógicos $\vee, \wedge, \rightarrow$, de forma a refletir o operador que relaciona as fórmulas A e B.*

Definição A.9 *Seja a o número que representa a fórmula A, a qual pode ser descrita como $\neg B$, sendo b o número que representa a fórmula B, então, $t(a)$ é o número de $B \rightarrow \perp$.*

Definição A.10 *$O(a)$ se, e somente se, a for o número da fórmula $\forall x B(x)$ e, então, $S[o(a), y]$ é o número de $B(0^{(y)})$.*

Definição A.11 *$E(a)$ se, e somente se, a for o número da fórmula $\exists x B(x)$ e, então, $S[e(a), y]$ é o número de $B(0^{(y)})$.*

Prova. Prova do lema A.3.1

Para cada k , a definição de verdade para $T_k(b)$ pode ser dada através de uma fórmula de Z, desde que satisfaça as condições abaixo:

a) Fórmulas sem quantificadores:

i) $T_0[D(m, a, n)]$ se, e somente se, $\exists y P[y, D(m, a, n)]$; ou

ii) $T_0[D(m, \bot, n)]$ se, e somente se, $\neg\exists y P[y, D(m, \bot, n)]$; ou

iii) Conjunção:

$$\exists x \exists y\{(x < D(m,a,n)) \wedge (y < D(m,a,n)) \wedge D(m,a,n) = t(x,y) \wedge [T_0(x) \wedge T_0(y)]\};$$

iv) Disjunção:

$$\exists x \exists y\{(x < D(m,a,n)) \wedge (y < D(m,a,n)) \wedge D(m,a,n) = t(x,y) \wedge [T_0(x) \vee T_0(y)]\};$$

v) Implicação:

$$\exists x \exists y\{(x < D(m,a,n)) \wedge (y < D(m,a,n)) \wedge D(m,a,n) = t(x,y) \wedge [T_0(x) \rightarrow T_0(y)]\};$$

vi) Negação:

$$\exists x \exists y\{(x < D(m,a,n)) \wedge D(m,a,n) = t(x,\bot) \wedge [T_0(x) \rightarrow T_0(\bot)]\};$$

b) Fórmulas com quantificadores

$T_{k+1}[D(m,a,n)]$ se, e somente se, $T_k[D(m,a,n)]$ ou,
$\{O[D(m,a,n)] \wedge \forall y T_{k+1}[D(m, S(o(a), y), n)]\}$ ou $\{E[D(m,a,n)] \wedge \forall y T_{k+1}[D(m, S(e(a), y), n)]\} ou$,
$\exists x \exists y\{(x < D(m,a,n)) \wedge (y < D(m,a,n)) \wedge D(m,a,n) = t(x,y) \wedge [T_{k+1}(x) | T_{k+1}(x)]\}$;

Pode ser vericado que $T_k(b)$ define verdade como em [29], logo $Con(Z^n)$ pode ser provada em Z. Note que $T_n(b)$ é uma definição de verdade apenas para o sistema Z^n e não para Z. □