UNIVERSIDADE ESTADUAL DE CAMPINAS

INSTITUTO DE FILOSOFIA E CIÊNCIAS HUMANAS

DEPARTAMENTO DE FILOSOFIA

PÓS-GRADUAÇÃO EM LÓGICA E FILOSOFIA DA CIÊNCIA

RAZÃO, COMPUTAÇÃO E SISTEMAS FORMAIS

Este exemplar corresponde à redação final da dissertação defendida e aprovada pela Comissão julgadora.
07/05/91

AUTOR: WAGNER DE CAMPOS SANZ

ORIENTADOR: PROF. DR. CARLOS ALBERTO LUNGARZO

Monografia apresentada como requisito para a obtenção do titulo de mestre

Lungarzo, Carlos, 1943

CAMPINAS

ABRIL DE 1991

AGRADECIMENTOS

A confecção desta dissertação só foi possível graças ao suporte financeiro fornecido pelo CNPQ, sob a forma de bolsa de estudos quando eu ainda era aluno da UFRGS no seu curso de Pós-Graduação em Filosofia. Mais tarde, quando por razões técnicas fois necessária a transferência para a Universidade de Campinas, a Fapesp; órgão do Estado de São Paulo, ofereceu suporte financeiro parcial a conclusão da mesma. Meus agradecimentos a estas duas instituições exemplares.

Gostaria ainda de agradecer aos professores e ao curso de Pós-Graduação em Filosofia da Universidade Federal do Rio Grande do Sul, onde comecei a desenvolver este trabalho, bem como aos porfessores e ao curso de Pós-Graduação em Filosofia da Universidade de Campinas onde consegui concluir o mesmo. Em particular gostaria de agradecer a assistência do meu professor orientador Dr. Carlos Alberto Lungarzo, como também o apoio dos professores Dra. Rejane Carrion e Prof. Antônio C. da Rocha Costa.

Gostaria ainda de agradecer a todos os meus colegas e amigos pela convivio agradável que me proporcionaram.

Por fim mas não menos, eu gostaria de agradecer o apoio e o carinho dedicados pela minha mulher Helena, sem a qual eu não seria metade do homem que sou, em felicidade e em peso.

`A Helena

INDICE

INTRODUÇÃO

A motivação deste trabalho vem de uma questão bastante polemizada nos nossos tempos: são os computadores capazes de se comportar inteligentemente? A Inteligência Artificial (I.A.) é uma disciplina que visa, entre outras coisas, construir máquinas e/ou sistemas que se comportem de modo inteligente.

Meu objetivo neste trabalho não é o de discutir ou analisar os diversos pontos de vista da polêmica em I.A., mas simplesmente procurar tecer, usando alguns conceitos filosóficos, algumas considerações que coloquem a questão em um quadro teórico mais amplo do que um quadro de conceitos puramente teórico-computacionais, pois a natureza da questão assim exige. Entretanto, para realizar tal tarefa será necessário articular alguns conceitos teórico-computacionais para sustentar a argumentação da conclusão. Diversos assuntos serão, então, abordados, assuntos estes pertencentes à áreas tão variadas quanto: fundamentos da teoria da computação, fundamentos da matemática, epistemologia etc.

O conceito inteligência é o centro da questão. Certamente qualquer pretensão de produzir uma réplica do comportamento inteligente passa antes pela exata compreensão do que vem a ser inteligência, o que significa comportamento humano inteligente. Não se iludam, eu não pretendo dar uma resposta a esta questão, meu objetivo aqui é muito mais limitado.

A questão da qual eu me ocuparei, e que aliás tem precedência temporal sobre a questão posta inicialmente, mas intimamente imbricada com a mesma, é: podemos fornecer um modelo mecânico da razão? O termo

"razão" é e será utilizado, para os propósitos deste trabalho, intercambiavelmente com o termo "inteligência". Se a resposta a esta pergunta é afirmativa então a primeira questão estará também resolvida, pois dado o modelo mecânico da razão poderemos utilizá-lo como modelo do comportamento inteligente para um computador.

O objetivo final da minha dissertação é o de mostrar a insustentabilidade de qualquer hipótese que afirme ser possivel dar um modelo mecânico completo do comportamento humano racional. Para alcançar tal objetivo eu tentarei articular uma série de conceitos filosóficos e teórico-computacionais fundamentais para a abordagem do problema.

O primeiro capítulo contém uma apresentação das hipóteses e pressupostos que, creio, são a base da pretensão de explicar o comportamento inteligente por meio de um modelo mecânico, o que eu chamo de hipótese mecanicista-formalista da razão. Eu acredito que a hipótese do sistema físico de símbolos expressa paradigmaticamente a hipótese mecanicista-formalista. Segue-se, portanto, uma apresentação do conceito de sistema físico de símbolos e da hipótese correspondente. Ao final deste primeiro capítulo eu procuro relacionar a hipótese mecanicista-formalista da razão com a doutrina do determinismo físico.

Os capítulos restantes servem para estabelecer conceitos teóricos relevantes para a argumentação conclusiva. As minhas metas com os capítulos restantes são: um exame da natureza dos computadores (são máquinas de sistemas formais); e um exame da natureza da tarefa de programar um computador.

O segundo capítulo trata do conceito de sistema formal. Ao final deste capítulo eu apresentarei uma definição formal deste conceito, útil as minhas necessidades nesta dissertação.

O terceiro capítulo é dedicado a uma tese básica para a construção

do argumento de conclusão. A tese que eu apresento e procuro sustentar é: os computadores são máquinas de sistemas formais.

O quarto capítulo contém uma apresentação acerca da atividade de programar, bem como a caracterização do que vem a ser um programa. O objetivo deste capítulo é o de revelar quais são as atividades canônicas na construção de um "software". Caso tais atividades sejam indispensáveis no processo, então a pretensão de construir uma máquina que tenha comportamento inteligente por via da programação implica que tais passos canônicos também devem estar presentes nesta construção. Tal argumento também oferece suporte para as conclusões finais.

Por fim a conclusão, onde eu pretendo organizar os argumentos que parecem implicar a não-sustentabilidade da hipótese mecanicista-formalista, com base nos teoremas de Gödel para a aritmética formalizada.

Um aviso final de utilidade para a leitura deste trabalho: a bibliografia usada e referida para e na confecção de cada capítulo está relacionada ao final dos mesmos.

CAPITULO I

RAZAO E MECANICISMO

A- O tema desta dissertação

Como eu havia dito na introdução a questão que motiva a discussão presente neste trabalho é: são os computadores capazes de um comportamento inteligente?

Volto a reiterar, não é meu propósito, nesta dissertação, discutir e analisar os termos da polêmica envolvidos nesta questão, nomeadamente aqueles sob o titulo de I.A., o que não significa que o tema tratado não seja polêmico. Talvez em um tempo futuro, se eu julgar conveniente, a polêmica possa ser tratada a partir das considerações aqui feitas. A questão que eu aqui quero discutir e que eu acredito estar intimamente relacionada com a questão anterior é: podemos dar um modelo mecânico da inteligência (ou razão)?

A pergunta que surge muito naturalmente é: como, afinal de contas, estão relacionadas estas questões? Vou tentar mostrar como estão relacionadas tais questões.

Primeiro, o conceito inteligência não pode ser dado como rigorosamente estabelecido, pois não possuimos critérios necessários e suficientes para tal. Assim sendo só há por enquanto uma via de acesso ao significado deste conceito, e esta via é a via da observação do comportamento humano que classificamos como inteligente. Em outras palavras, por hora só podemos utilizar a palavra inteligência, ou razão,

dentro do contexto do comportamento humano, é a única maneira de dar sentido a tal conceito. Logo se perguntamos pela possibilidade de um computador apresentar comportamento inteligentemente, devemos, necessariamente, perguntar se um computador poderia comportar-se como um ser humano o faz, desde que, é claro, tal comportamento humano pudesse ser classificado de inteligente, ou racional.

Segundo, um computador é uma máquina, e ser uma máquina significa, entre outras coisas, acredito eu, ser passível de descrição mecânica. Bem, mas o que vem a ser uma descrição mecânica? Não sei se tenho uma resposta adequada, mas, mais adiante, eu tentarei dissertar um pouco sobre este ponto. De todo modo, como eu tentarei mostrar no terceiro capítulo, os computadores são máquinas de sistemas formais. Com esta afirmação eu quero apenas mostrar que toda descrição da ação de um computador pode ser feita através de um sistema formal. O conceito de sistema formal usado nesta dissertação será discutido e caracterizado no segundo capítulo.

Podemos concluir que a possibilidade de responder afirmativamente a questão inicial depende de uma resposta afirmativa à seguinte questão: podemos replicar o comportamento humano inteligente em um computador? Isto porque a via de acesso ao significado da palavra "inteligência" pressupõe, ao menos preliminarmente, a observação do comportamento humano classificado como inteligente. Mas responder afirmativamente a esta questão equivale a responder afirmativamente a seguinte questão: podemos fornecer um sistema formal suficiente para o comportamento inteligente, ou racional? A hipótese que representa uma resposta afirmativa a esta questão eu chamo de hipótese mecanicista-formalista da razão.

A tese da replicação do comportamento racional tem o seu fascínio. Este fascínio traduz, creio eu, um desejo de auto-conhecimento, o desejo

de compreender em que consiste, no final das contas, a inteligência e a razão humanas. Naturalmente alguém poderia afirmar que conhece o "mecanismo" da razão caso esse alguém fosse capaz de replicar intencionalmente tal "mecanismo".

No meu entender, a hipótese que sustenta a possibilidade da replicação baseia-se em dois pressupostos: primeiro, raciocinar é o mesmo que calcular; segundo, calcular é o mesmo que manipular símbolos por meio de regras. E´ mais ou menos este o ponto de vista de Haugeland ([1] pág 23), que afirma ser Hobbes o avô de I.A. por causa da seguinte frase: "By raciocination, I mean computation". Mais especificamente, Haugeland sustenta que as principais teses de GOFAI ("Good Old Fashioned Artificial Intelligence") são: "primeiro, nossa habilidade de lidar com as coisas inteligentemente é derivada da nossa capacidade de pensar sobre elas razoavelmente; segundo, nossa capacidade de pensar sobre as coisas razoavelmente resume-se a uma faculdade de manipulação interna, automática, de símbolos" ([1] pág 113).

A hipótese de que existe um sistema formal suficiente para o comportamento racional não explica propriamente como deve ser este sistema formal, mas sustenta a viabilidade de um certo tipo de explicação teórica, justamente a explicação mecanicista-formalista do comportamento racional.

Qual a origem de tal hipótese? A resposta não é fácil, talvez as origens sejam tão antigas quanto a filosofia. Entretanto o que eu posso fazer é apresentar um fato que mostra que a hipótese ao final das contas não é absurda, tal fato pode até mesmo ter sido um provável motivação original.

Quando fazemos um cálculo com números naturais usamos um algoritmo

para trabalhar com os numerais que representam os números naturais. No caso de um cálculo de divisão entre dois números naturais procedemos segundo regras que a nossa professorinha primária nos ensinou. Como usualmente representamos os números por meio de algarismos arábicos as regras para efetuar a nossa continha de dividir são nada mais nada menos que regras para a manipulação de expressões compostas de algarismos arábicos, ou seja regras para manipulação de símbolos. Mas estas regras são adequadas unicamente para números representados por algarismos arábicos. Para tomar ciência deste fato basta tentar fazer a mesma divisão com os mesmos números só que agora representados por meio de algarismos romanos. Logo percebe-se que não podemos usar as mesmas regras! Há claramente uma inter-dependência entre as noções de divisão, representação de números por símbolos e manipulação destes símbolos por meio de regras. Ora se realmente a divisão é pensada como uma noção definida por meio de regras e símbolos manipuláveis por estas regras então pareceria lícito pensar que esta e outras operações que podemos realizar dependem fundamentalmente das regras e dos símbolos envolvidos na noção.

Se tomarmos a essência da racionalidade como a capacidade de calcular, ou seja se raciocinar é o mesmo que calcular, e se calcular pode ser compreendido como manipular símbolos segundo regras então parece perfeitamente lícito esperar que a hipótese da replicação seja verdadeira.

Particularmente, penso que somos capazes de processar símbolos segundo regras, o que não significa necessariamente que tudo o fazemos inteligentemente possa ser reduzido a processar símbolos por meio de regras. Mesmo porque a noção de divisão, por exemplo, não necessariamente precisa ser pensada a partir de uma representação

simbólica. Por exemplo, na distribuição de laranjas para um grupo de pessoas fazemos uma operação que nada tem de simbólico e que me parece merece o nome de divisão!

Bem, mas eu tenho repetidas vezes falado em comportamento racional. Usualmente a palavra utilizada no contexto da hipótese de relicação é a palavra "mente". Dado um certo pudor particular eu prefiro não usar esta palavra. Na conclusão deste trabalho eu fornecerei algumas razões deste pudor. De qualquer modo, acredito que é necessário caracterizar, ao menos parcialmente, o que eu entendo por comportamento racional. Tal caracterização é muito importante para a sequência de argumentos da conclusão.

Falar com sentido certamente é um dos comportamentos que podemos chamar de racional, mas principalmente ter a capacidade de distinguir o verdadeiro do falso através de nossos julgamentos é inquestionavelmente um comportamento racional. Evidentemente não podemos afirmar que os seres humanos conseguem sempre distinguir o verdadeiro do falso, tanto quanto não podemos afirmar que alguém relativamente incapaz de distinguir o verdadeiro do falso não possa ser de algum modo inteligente. Tudo depende da definição do conceito "inteligência". O que eu quero de fato apontar é que a capacidade de distinguir o verdadeiro do falso por ser um comportamento racional deve ser passivel de ser capturado por um modelo mecanicista-formalista da razão. Em outras palavras, um sistema formal adequado como modelo da razão deve ser capaz de distinguir o verdadeiro do falso tanto quanto algum de nós pode fazê-lo em bases racionais.

B- O Caso do Sistema Físico de Símbolos

Nesta secção eu apresentarei a hipótese do sistema físico de símbolos, de Newell e Simon, com dois objetivos: primeiro, mostrar que a questão posta na secção anterior realmente se refere a um questionamento existente, pois a hipótese do sistema físico de símbolos é a tese "GOFAI" por excelência; segundo, apresentar alguns pontos básicos importantes para a próxima secção.

Em uma Turing Lecture ([3]) os autores apresentam suas opiniões sobre a ciência da computação sustentando, principalmente, que ela é uma disciplina empírica. Dois são os tópicos principais desta conferência: a noção de sistema simbólico e a noção de busca heurística. Os dois temas são de interesse para esta dissertação, contudo eu vou me ater mais ao primeiro tema nesta apresentação, a noção de sistema físico de símbolos. Segue-se portanto uma apresentação deste assunto, calcada no referido artigo da Turing Lecture.

Segundo Newell e Simon, os símbolos estão na raiz da ação inteligente, que por sua vez é o tópico primário de I.A. Para os autores, não há propriamente um princípio da inteligência, entretanto isto não significa que não existam requisitos estruturais para distinguir o que é inteligência do que não é. Um destes requisitos é a habilidade de manipular símbolos.

Um sistema físico de símbolos consiste de um conjunto de entidades chamadas símbolos, que são padrões físicos que podem ocorrer como componentes de outros tipos de entidades chamadas expressões. Logo, uma expressão é composta de um certo número de símbolos (ou "tokens")

relacionados de algum modo físico. Em qualquer instante de tempo o sistema contém uma coleção destas expressões. Ao lado destas expressões o sistema também contém uma coleção de processos que operam sobre as expressões para produzir outras expresões, são eles: processos de criação, modificação, reprodução e destruição. Um sistema físico de símbolos é uma máquina que produz através do tempo uma coleção de estruturas de símbolos. Tal sistema existe em um mundo de objetos mais amplo do que justamente este sistema de expressões simbólicas.

O adjetivo "físico" denota aqui duas importantes caracteristicas: primeiro, tais sistemas claramente obedecem leis da física, eles são realizáveis por sistemas gerados por engenharia composto de componentes também gerados por engenharia ([3] pág. 116); segundo, embora o uso do termo "símbolo" pré-figura nossa interpretação (o símbolo manipulado por nós), os sistemas de símbolos não estão restritos aos, digamos, sistemas humanos.

Duas noções, segundo os autores, são essenciais a esta estrutura de expressões, símbolos e objetos:

a) designação - uma expressão designa um objeto se, dada a expressão, o sistema pode ou afetar o objeto, ele mesmo, ou se comportar de modo dependente do objeto;

b) interpretação - o sistema pode interpretar uma expressão se a expressão designa um processo e se, dada a expressão, o sistema pode levar adiante o processo.

Não estou completamente seguro quanto ao significado destas duas noções. Penso que por interpretação os autores querem simplesmente afirmar que uma regra ou um programa enquanto expressões, podem ser efetivamente realizados, ou seja interpretados, por um processo. Tal processo seria, nesse caso, designado por tal expressão que é por seu

turno um programa ou uma regra.

Um sistema físico de símbolos é um exemplo de uma máquina universal. Logo a hipótese do sistema de símbolos implica que a inteligência será efetivada por um computador universal.

Definido o que vem a ser um sistema físico de símbolos, Newell e Simon passam a postulação de uma hipótese, que segundo eles tem a carcaterística de ser uma lei qualitativa, em outros termos, uma lei que afirma a possibilidade de uma certa explicação científica, mas que não explica propriamente o fenômeno, e que diz respeito ao que se pode considerar como requisito estrutural para a inteligência:

* Hipótese do Sistema Físico de Símbolos: Um sistema físico de símbolos tem os meios necessários e suficientes para a ação inteligente geral. Por "necessário" os autores entendem que qualquer sistema que exiba ação inteligente geral mostrar-se-á, sob análise, ser um sistema físico de símbolos. Por "suficiente" eles entendem que qualquer sistema físico de um tamanho suficiente pode ser organizado posteriormente para poder apresentar inteligência. Pelos termos "ação inteligente geral" eles desejam indicar o mesmo escopo da inteligência tal como ela é vista na ação humana: tal que qualquer comportamento em uma situação real apropriada aos fins do sistema e adaptiva as demandas do meio-ambiente podem ocorrer, dentro de alguns limites de velocidade e complexidade.

Uma das raízes desta hipótese, no entender dos seus mentores, está nos programas de formalização da lógica de Frege e de Whitehead e Russell, pois as características de jogo simbólico da lógica remonta a estes programas. "Logic, and by incorporation all of the mathematics, was a game played with meaningless tokens according to certain purely syntactic rules" ([3] pág. 117). Ao que parece os autores defendem uma

formalismo da matemática bastante forte.

Segundo os autores, a hipótese do sistema físico de símbolos é uma hipótese empírica. Dizem eles que não conhecem nenhuma maneira de demonstrar a conecção entre os sistemas de símbolos e a inteligência por métodos puramente lógicos. Logo, se falta tal demonstração então os fatos, ou evidências, devem servir de referência.

Sobretudo, eles afirmam explicitamente, e isto nos interessa muito aqui nesta dissetação, que: "The symbol system hypothesis implies that the symbolic behavior of man arise because he has the characteristics of a physical symbol system." ([3] pág. 119).

Contudo, a caracterização do que é um comportamento inteligente ultrapassa a hipótese inicial do sistema físico de símbolos. Em uma hipótese de segundo nível os autores procuram especificar mais precisamente que tipos de sistemas de símbolos são adequados. Esta é a hipótese da procura heurística.

Antes de formular tal hipótese convém mostrar qual a sua relevância. A palavra chave para entender a relevância da procura heurística é a palavra "problema". Para Newell e Simon, a capacidade de resolver problemas é o indice mais básico da existência de inteligência. Este é um paradigma para a I.A, um sistema físico de símbolos é inteligente se ele é capaz de resolver problemas.

A hipótese da procura heurística afirma: as soluções dos problemas são representáveis por estruturas de símbolos, e um sistema de símbolos exerce a sua inteligência para resolver o problema por procura heurística, ou seja, gerando e progressivamente modificando estruturas de símbolos até que seja gerada uma estrutura de símbolo solução do problema. O sistema de símbolos tem inteligência na razão inversa da quantidade de procura que ele apresenta até achar uma solução.

Ainda que a questão heuristica seja muito relevante, dados os propósitos estabelecidos nesta dissertação acerca dela eu não farei mais nenhuma apresentação ou referência.

Penso que o sistema físico de símbolos e a hipótese correspondente lembram muito a apresentação de uma mecânica da razão, exceto que no lugar de símbolos estão as idéias, feita pelo empirismo clássico (Hume por exemplo). A inteligência para Newell e Simon é derivada de uma série de processos e símbolos sobre os quais atuam estes processos, tal como para os empiristas existem uma série de relações que se podem estabelecer entre as idéias, e raciocínios que se podem fazer usando as idéias e as referidas relações. A meu ver tal semelhança tem raízes profundas.

C- Razão, Mecanicismo e Determinismo

Como o leitor atento percebeu, na secção anterior eu coloquei em negrito uma passagem. Segunda esta passagem, extraída do texto de Newell e Simon, os sistemas físicos de símbolos claramente devem obedecer as leis da física, pois estes sistemas físicos são nada mais nada menos que máquinas universais. A este tipo de hipótese que afirma ser possível explicar a inteligência por meio de um sistema de símbolos eu chamei de mecanicismo-formalista, pois o sistema físico de símbolos é nada mais nada menos que uma máquina de manipulação formal automática de símbolos. Eu acredito que posso produzir uma caracterização melhor deste conceito de manipulação automática de símbolos. Ao chamar um computador de máquina de sistemas formais, conforme o terceiro capítulo, eu acredito poder caracterizar bem por um lado a existência de um agente capaz de ser tratado em termos puramente mecânicos, logo um sobre o qual cabe perfeitamente qualquer caracterização por meio de leis físicas; e por outro lado a existência de uma descrição do comportamento do agente, ou seja a ação descrita por um sistema formal.

Nesta secção eu pretendo articular a doutrina do determinismo físico com o mecanicismo-formalista da razão, a partir das idéias de J.R Lucas, expostas no livro "The Freedom of the Will" ([2]). Isto se deve as seguintes razões: primeiro, a estrutura dos argumentos que eu utilizarei na conclusão para rejeitar o mecanicismo-formalista da razão tem a mesma estrutura dos argumentos que Lucas usa para rejeitar o determinismo físico; segundo, parece haver uma conexão doutrinária entre a tese que eu chamo de tese mecanicista-formalista da razão com a tese do determinismo

físico. No bojo desta discussão creio que poderemos compreender o que significam os termos "ser mecânico" e a que objetos tais termos podem ser aplicados.

O determinismo físico é uma doutrina que afirma que os estados futuros de um objeto do mundo físico estão necessariamente determinados. O determinismo ve o mundo como um conjunto de elementos inter-relacionados causalmente. As caracteristicas de um determinado elemento dependem necessariamente, em cada instante de tempo, das caracteristicas de todos os outros elementos que possam afetar causalmente o elemento em questão nos instantes anteriores ao instante considerado. A categoria de causalidade ultrapassa assim o nível meramente explicativo e passa realmente a impor uma ordem sobre as coisas do mundo. Para saber qual o estado de um determinado objeto, ou até mesmo do mundo físico, em um tempo futuro tudo o que precisamos ter é: uma descrição do estado atual, ou inicial; e um conjunto de leis que seja completo e que determine o comportamento dos elementos constituintes da descrição do estado. Segundo Lucas a doutrina do determinismo coloca problemas a ação e ao julgamento moral. No seu livro ele procura mostrar o determinismo como inválido usando argumentos semelhantes aos argumentos usados na conclusão desta dissertação deixando assim espaço para a possibilidade da ação moral.

Segundo Lucas o determinismo físico foi concebido a partir de um casamento da física newtoniana com a metafísica lockeana. Para que o determinismo fosse efetivo como doutrina duas condições deveriam ser cumpridas: primeiro, as descrições de estado deveriam ser completas; segundo, a variável tempo deveria entrar de alguma forma nas leis que regem os estados e seus elementos. Segundo o autor, a física newtoniana cumpre perfeitamente este papel pois os elementos da doutrina newtoniana são particulas puntuais, com massa, impenetráveis e qualitativamente

idênticas, e as qualidades importantes a serem conhecidas acerca destas particulas são a sua posição e o seu momentum. Deste modo poderíamos conhecer completamente um estado se conhececemos a posição e o momentum das particulas pertencentes ao estado. Por outro lado as leis newtonianas introduzem dependência funcionais entre as posições e os momenta das particulas de um determinado sistema, tal que estas dependências funcionais tem como variável independente a variável tempo. Entretanto, só a física newtoniana não é suficiente para engendrar o determinismo. E' ainda necessário uma doutrina que sustente a reducibilidade de todas as qualidades de um sistema a qualidades simples e básicas. Este é o papel desempenhado pela metafísica lockeana acerca das substâncias e das qualidades primárias e secundárias. Para Locke as qualidades secundárias devem ser explicadas em termos das qualidades primárias. Resultado: na tese determinista todas as qualidades deveriam pode ser descritas em termos da posição e do momentum das partiículas elementares, entendidas então como as qualidades primárias as quais deveriam se reduzir todas as outras. A metafísica lockeana passa então a garantir que as as descrições de estado da física newtoniana são completas. "According to Locke, if we know the positions and momenta of the fundamental particles, then we should, if we knew enough science and were clever enough, know everything, in every respect" ([2] pág 87).

Ter uma descrição do estado inicial de um sistema, logo um conjunto de sentenças estabelecendo este estado inicial, e ter um conjunto de leis, logo um conjunto de sentenças universais acerca dos elementos da descrição de estado do sistema é o mesmo que possuir um cálculo e os elementos iniciais de um cálculo tal que os estados futuros possam ser calculados por dedução. Mas ter um cálculo neste sentido é o mesmo que

possuir um sistema formal, um cálculo logístico no dizer de Lucas, que permita obter previsões dos estados no tempo futuro. Esta é a ligação do determinismo físico com o conceito de sistema formal. Por um lado enquanto os estados do mundo geram os estados futuros por causação, nós podemos gerar a descrição dos estados futuros a partir da descrição atual e de leis completas por meio de dedução.

A concepção doutrinária do determinismo baseia-se numa presmissa do seguinte teor: toda as leis científicas são leis de regularidade. Por seu turno as leis de regularidade são nada mais nada menos que relações causais que podem ser quantificadas, trocando em miúdos, leis de dependência funcional. Para que a tese da possibilidade de determinar o estado futuro de um sistema possa ser mantida é necessário que essas leis que introduzem uma dependência funcional entre os elementos do estado introduzam uma dependência funcional calculável.

Se observarmos bem podemos concluir que o fisicalismo dos modernos empiristas do círculo de Viena alia-se, ao menos em parte, a esta tradição do determinismo físico, pois outra não é a pretensão do fisicalismo senão a de sustentar que todos os conhecimentos científicos podem ser reduzidos a uma base comum: a base física.

Assim, a tese de que a racionalidade pode ser explicada em termos mecânicos é uma tese que repousa em última instância na doutrina do determinismo físico e da redutibilidade dos conhecimentos a base física. A tese da redutibilidade a base física tem a sua origem na pressuposição de que toda e qualquer qualidade secúndária deve ser explicada em termos de qualidades primárias. Como a posição e o momentum das particula elementares fornecem a descrição completa de um estado físico, logo qualquer outra qualidade, como o comportamento humano, deve poder ser explicada em termos destas qualidades primarias, ou ao menos deve poder

ser explicada em termos de qualidades físicas redutíveis as qualidades primárias, visto que em última instância somos constituídos de particulas elementares.

A força do determinismo físico enquanto doutrina é derivada do sucesso da física nos últimos trezentos anos como disciplina científica. Este apelo provém do fato de serem as explicações de regularidade da física um paradigma de explicação racional.

Eu acredito que atribuir a qualidade "mecânico" a um certo sistema significa, então, identificar completamente os estados do sistema e dar um conjunto de leis completo acerca do sistema, leis estas que são basicamente leis físicas de regularidade, de causa e efeito.

A hipótese mecanicista-formalista é uma que acredita na viabilidade de fornecer um sistema formal adequado que exiba ação inteligente geral. No caso do sistema físico de símbolos de Newell e Simon estes afirmam explicitamente que um sistema de símbolos é um mecanismo que pode ser descrito por leis físicas.

A meu ver, esta ligação entre símbolos e máquinas dá-se exatamente porque podemos construir uma máquina de tal modo que o seu comportamento seja descrito por um sistema formal. Esta é a essência da máquina de Turing. Nela os estados da máquina são adequados para representar símbolos e processos sobre símbolos!

A afirmação da existência de um parentesco entre o modelo mecânico do comportamento racional com a doutrina do determinismo físico não é inescapável. Alguém pode simplesmente afirmar que raciocinar é calcular e calcular é manipular símbolos por meio de regras, e que isto não tem necessariamente relação com leis físicas. Para que tal parentesco ficasse absolutamente estabelecido seria necessário mostrar que o conceito

"sistema formal" é de algum modo equivalente ao conceito "ser mecânico". Neste caso o termo "mecanicista-formalista" seria redundante. De todo modo uma das grandes invenções do nosso século foi a máquina de Turing, pois Turing mostrou que uma máquina é capaz de processar símbolos para isso utilizando um sistema físico que possua estados, correlacionando estes estados com os símbolos processados e com as regras de processamento.

D- Bibliografia

[1] Haugeland, J.; ARTIFICIAL INTELIGENCE: THE VERY IDEA, MIT Press, 1987

[2] Lucas, J.R.; THE FREEDOM OF THE WILL, Claredon Press Oxford, 1970

[3] Newell, A. e Simon, H.A.; COMPUTER SCIENCE AS EMPIRICAL INQURY: SYMBOLS AND SEARCH, Communications of the ACM, v. 19, nro. 3, págs. 113-126, 1976

CAPITULO II

SISTEMAS FORMAIS

A- Introdução

No capítulo anterior eu focalizei o tema desta dissertação. Ao fazê-lo pudemos notar que o conceito de sistema formal é central para o entendimento da questão da qual me ocupo aqui.

Neste capítulo o conceito de sistema formal será tratado sob os aspectos que venham a ser relevantes para a dissertação como um todo. Para começar, o conceito será definido um tanto intuitivamente e os seus aspectos que creio essenciais serão destacados. Em segundo lugar, será apresentado um pequeno histórico do uso do conceito dentro do formalismo hilbertiano. Em terceiro lugar, serão articulados os conceitos de sistema formal e linguagem artificial. Em quarto lugar, a discussão do problema filosófico da significação. Em quinto lugar, desde o ponto de vista de Smullyan, o conceito de sistema formal será definido precisamente. Esta definição será tomada por base quando, posteriormente, eu fizer referência ao conceito de sistema formal.

Com este capítulo eu pretendo cobrir toda a discussão que seja relevante para o uso do conceito de sistema formal dentro desta dissertação.

B- Sistemas Formais, Caracterização Preliminar

No sentido aqui pretendido um sistema formal é composto por: primeiro, um conjunto de símbolos no máximo contável; segundo, uma ou algumas classes de expressões obtidas a partir de regras de formação, de tal modo que cada classe é um subconjunto de todas as sequências finitas sobre o conjunto de símbolos, podendo ser igual, sendo que aos elementos deste conjunto eu os denomino palavras; terceiro, um subconjunto recursivo de uma das classes de expressões chamados de axiomas (ou de esquemas de axiomas); por fim um conjunto finito de relações poli-árias, tal que a aridade de cada relação é finita, sobre o conjunto das expressões, chamados de regras de transformação. Há, entretanto, sistemas formais degenerados que não possuem nem axiomas, nem regras de transformação, por exemplo as gramáticas formais, logo toda gramática formal é um sistema formal no sentido útil aos propósitos desta dissertação.

Ora, um sistema formal parece ser, dada a descrição anterior, um jogo onde as peças elementares são símbolos que de alguma maneira elementar possam ser distinguidos entre si. Dada a capacidade de distingui-los torna-se possível agrupá-los segundo regras de formação. O conceito de sistema formal a que eu aqui me refiro é um no qual a operação fundamental é a da concatenação de símbolos distintos e distinguíveis entre si. Logo, as regras de formação para expressões informam quais as concatenações de símbolos são admissíveis, e quais são as diferentes classes de expressões admissíveis. Para esta dissertação só serão considerados sistemas formais aqueles que possam ser representados,

em outras palavras somente os sistemas "escreviveis" serão tomados como sistemas formais.

Neste sentido de sistema formal, se existe um processo efetivo de distinguibilidade elementar para os símbolos é de se esperar que, ao menos em possibilidade, o processo de distinguibilidade de segundo nível, ou seja aquele de distinção entre expressões, seja efetivo, na medida em que a geração de novos padrões pelas regras de formação seja efetivo. Tais classes de expressões formam o que podemos chamar de uma linguagem, ou conjunto de palavras. Se deste conjunto de palavras destacamos algumas, e, além disso, estabelecemos algumas regras que permitam a partir deste conjunto de palavras destacadas obter outras palavras, estamos frente ao que se chamaria de um conjunto recursivamente definido a partir de palavras iniciais e regras. Em muitos destes sistemas as palavras poderão ser sentenças, mas isto não é verdadeiro para todos os sistemas que eu estou a considerar aqui. Obvio está que as novas palavras geradas, por meio das regras, ditas regras de transformação, devem pertencer ao conjunto de expressões, ou seja a linguagem.

O exame de um caso pode tornar as coisas um pouco mais claras. Por exemplo um sistema formal com uma regra infinitária não é propriamente um sistema formal no sentido aqui pretendido, pois a aplicação da regra dependeria da possibilidade de identificar, e portanto escrever, uma infinidade de premissas. Como não podemos manipular nenhuma quantidade infinita de expressões um sistema qúe possua uma regra infinitária não será considerado um sistema formal.

Convém ressaltar que os elementos mais importantes de um sistema formal são as suas regras. Sistemas formais com símbolos diferentes mas com as mesmas regras, de tal modo que se possa estabelecer,

adequadamente, uma relação de equivalência entre os símbolos e expressões de um sistema e do outro, são, no fundo, o mesmo sistema formal. Ou seja, eles tem as mesmas propriedades formais.

Os axiomas também tem um papel relevante, mas eles podem ser adequadamente pensados como regras de transformação.

Esta caracterização inicial do conceito de sistema formal tem o objetivo de delimitar, ainda que não precisamente, o objeto de discussão para os próximos capítulos e secções. Contudo, ao final deste capítulo eu apresentarei uma definição exata do conceito de sistema formal segundo o sentido de Smullyan, que é exatamente o sentido pretendido nesta dissertação. Logo, serão tratados como sistemas formais, segundo o sentido aqui pretendido, todos aqueles que tenham a forma de um sistema formal elementar (ver secção E deste capítulo) ou que possam ser escritos como um.

C- Histórico

O conceito de sistema formal foi postulado por Hilbert com objetivos precisos dentro do contexto de fundamentação da matemática. Hilbert pretendia mostrar uma prova de consistência da matemática clássica de uma maneira finitista, ou seja empregando somente objetos intuitivamente concebíveis e processos efetivamente realizáveis.

O sistema formal, ou inventário de fórmulas, era construido de modo tal que a cada proposição da matemática correspondesse uma fórmula do sistema formal. Assim a prova de uma proposição matemática era transformada em uma sequência de passos onde as fórmulas do inventário eram manipuladas segundo regras sintáticas bem estabelecidas. Como o próprio Hilbert salientava, os símbolos e fórmulas deste inventário eram completamente destituídos de significado de tal modo que a manipulação não fazia nenhuma referência ao significado das expressões, sendo puramente sintática ([8] pág. 381).

Portanto, provar a consistência da matemática clássica, aquela que continha a análise como parte, se traduzia em provar um teorema sobre as provas dentro do sistema formal, uma metaprova mostrando ser impossível obter uma fórmula correspondente a proposição $0 \neq 0$.

Hilbert afirmava que uma explicação matemática era perfeita se ela tivesse duas caracteristicas: a primeira, devia explicar alguma coisa relevante; a segunda, tal explicação não poderia aportar nenhuma inconsistência ao edifício da matemática ([8] pág. 370). Em particular o objetivo de Hilbert era o de mostrar que a hipótese do contínuo cantoriana se ajustava neste papel.

Como, para ele, as proposições matemáticas deviam ter um elemento intuível, no sentido kantiano ([8] pág. 376), para poderem ser ditas verdadeiras ou falsas, e como a intuição não poderia capturar objetos infinitos, tratava-se de mostrar, então, que o uso do infinito na matemática desempenhava um papel relevante: completava o edifício da matemática, e não aportava nenhuma inconsistência a este edifício. Em particular, todas as leis lógicas aristotélicas eram válidas para este edifício, o da matemática clássica, e, mais especificamente, também era válida a lei do terceiro excluído. As proposições que faziam, de alguma forma, referência ao infinito eram chamadas de proposições ideais.

O cálculo logístico hilbertiano é uma linguagem de sinais sem propósitos de comunicação capaz de representar proposições matemáticas em fórmulas e expressar inferências lógicas através de processos formais. Deste modo obtem-se um inventário de fórmulas formadas de símbolos lógicos e matemáticos que seguem umas das outras de acordo com regras definidas. Destas fórmulas algumas correspondem aos axiomas, e as inferências conteudísticas correspondem regras de acordo com as quais as fórmulas seguem-se umas das outras, ou seja as inferências são substituídas por regras de manipulação de símbolos. ([8] pág. 381)

A metateoria por meio da qual Hilbert pretendia dar uma demonstração da consistência da matemática ele chamava de metamatemática, e nesta somente os objetos intuitivamente concebíveis e processos efetivamente realizáveis deveriam ser utilizados.

Já na metade do século XX uma tendência muito forte tornou-se à atitude filosófica predominante dentro da matemática: a atitude formalista forte. Ela descende do formalismo Hilbertiano, mas não é filha direta. Esta tendência tem mais parentesco com o positvismo lógico do Círculo de Viena. Segundo Davis e Hersh ([5]), os formalistas no segundo

sentido, concebem a matemática como um jogo de dedução lógica. A matemática é entendida por estes formalistas como a ciência da prova rigorosa. Daqui para diante eu vou designar tal atitude de "formalista-positivista".

A meu ver a atitude formalista-positivista tem uma forte influência sobre a hipótese mecanicista-formalista da razão. As linhas desta influência podem ser assim traçadas: se resolver um problema é indício de inteligência, e se a resolução de um problema matemático se resume a uma prova formal, que é nada mais nada menos que a manipulação de símbolos por meio de regras, então a inteligência e a razão podem ser efetivados mecanicamente desde que uma máquina possa adequadamente processar símbolos a partir de regras.

D- Sistemas Formais e Linguagens Artificiais

Todo sistema formal que possua regras de transformação define um cálculo, onde os elementos do cálculo são expressões definidas por intermédio das regras de formação. Como salienta Carnap, os cálculos mais importantes são as linguagens ([1] pág 5).

Dentro das linguagens construídas as regras de formação definem quais as expressões que a linguagem possui, em particular elas definem o que é uma sentença. As regras de transformação, por sua vez, nos permitem operar, de algum modo, com as expressões definidas pelas regras de formação. Através de uma regra de transformação podemos expressar uma regra de inferência lógica, bem como podemos expressar postulados.

As regras lógicas de dedução podem ser transformadas em regras de sintaxe. Neste caso as regras de inferência passam a ser regras que dependem unicamente para a sua aplicação da forma das expressões.

As linguagens construídas servem basicamente para expressar teorias de um modo formal. As regras de formação determinam a forma das expressões, como eu já havia observado, e as regras de transformação nos permitem formular os axiomas (ou postulados) da teoria, bem como as leis lógicas de inferência. Podemos, então, representar formalmente, sem referência nenhuma ao significado das sentenças, uma teoria e um aparato lógico sintático para dedução de todos os teoremas da teoria.

Um exemplo: a) símbolos - {0, ′, =};

b) regras de formação -

b.1) termos -

0 é um termo (constante)

se X é um termo então X' é um termo

b.2) fórmulas -

se Y e Z são termos então Z=Y é uma fórmula

c) regras de transformação -

c.1) axioma - 0=0

c.2) regra de dedução - se X=Y então X'=Y'

Ao estudo da estrutura das linguagens, enquanto cálculo, Carnap chamava de sintaxe lógica da linguagem ([1] pág 1). Para ele esta era uma teoria analítica das formas do cálculo. Dentro da sintaxe lógica da linguagem importava investigar tudo o que dizia respeito as definições das expressões, e as consequências de tais definições, mas unicamente sob o aspecto da sua forma. Logo, o projeto carnapiano, entendido sob este ponto de vista, era mais amplo que o projeto hilbertiano, que só se ocupava de mostrar a consistência como propriedade da matemática clássica.

Numa primeira etapa do pensamento carnapiano a filosofia da ciência se resumia a sintaxe lógica da linguagem da ciência. Em uma etapa posterior o autor passa a admitir que a filosofia da ciência devia incluir as questões semânticas e pragmáticas dentro deste quadro de estudo da linguagem da ciência. Em particular, o autor passou a sustentar que a definição de denotação e a definição de verdade para os elementos da linguagem podem ser feitos dentro de uma outra linguagem, a metalinguagem, de tal forma que na metalinguagem podemos falar das "coisas" denotadas e das expressões da linguagem objeto. Para Carnap é possível falar acerca do significado das expressões de uma linguagem dentro de uma linguagem. No sistema semântico são estabelecidas, por meio de regras semânticas, as condições necessárias e suficientes para determinar a verdade das expressões da linguagem. Segundo ele, na fase de

"Introduction to Semantics" ([4]), conhecer o significado de uma expressão era o mesmo que conhecer as condições de verdade da expressão, logo o sistema semântico definia o significado das expressões da linguagem.

O projeto carnapiano está intimamente ligado ao positivismo lógico do Círculo de Viena. O positivismo sustentava a necessidade de unificar a ciência sustentando a necessidade de formalização da linguagem da ciência. Como eu havia notado na secção anterior a filosofia formalista-positivista da matemática descende de tal concepção. Segundo as concepções positivistas a matemática era meramente uma linguagem para expressão das outras ciências.

Os empiristas do círculo de Viena procuravam por meio de um grande projeto unificar a ciência, e tal unificação passava pelo uso de uma linguagem exata e universal para expressar todas as ciências. Por outro lado o paradigma básico de ciência para os positivistas de modo geral era a física. Não surpreende, então, o fato de alguns destes positivistas pregarem a unidade da ciência justamente por meio de um fisicalismo. Ou seja, dados conceitos físicos básicos, todos os outros conceitos científicos de algum modo poder-se-ião reduzir a conceitos físicos. Assim uma linguagem formal que expressasse exatamente estes conceitos físicos básicos e suas relações seria a base da unidade da ciência pois todos os conceitos científicos deveriam poder ser definidos de algum modo a partir destes conceitos básicos e portanto serem expressados por esta linguagem da ciência.

E- A Questão do Significado

Uma das questões filosóficas mais importantes no contexto desta dissertação é a questão da significação. As dificuldades inerentes a questão são muitas, o tema é muito complexo. Nesta secção eu pretendo tratar parcialmente desta questão, apresentando uma relação possível entre uma linguagem formalizada, ou seja um sistema formal, e o significado dos elementos desta linguagem.

Usar expressões para transmitir e entender significados é provavelmente uma das características da nossa inteligência. Portanto, se estamos examinando a tese mecanicista-formalista da inteligência nesta dissertação então é muito justo que examinemos a questão da significação. Ora, como vimos a tese mecanicista-formalista se resume a afirmar que existe um sistema formal suficiente para dar conta do fenômeno da razão e da inteligência, logo como consequência desta tese é forçoso concluir que a significação, de algum modo, deve poder ser compreendida a partir do conceito de sistema formal. Para admitir que a significação possa ser explicada a partir do conceito de sistema formal deve-se admitir que as regras do sistema formal são suficientes para gerar o fenômeno da significação. Passo a examinar, não exaustivamente, esta tese que eu chamo de tese sintaxista da significação.

Fornecer um sistema semântico para algum cálculo, ou sistema formal, se reduz a fornecer regras de interpretação em uma metalinguagem onde possamos falar dos elementos do cálculo e dos objetos denotados por estes elementos do cálculo. Mas esta metalinguagem é tal que podemos compreender o significado das suas partes, e através da significatividade

inerente a esta metalinguagem são estabelecidos os significados dos elementos da linguagem objeto. A meu ver, o uso da metalinguagem para falar acerca do significado dos elementos da linguagem objeto é legitimo, enquanto o papel desta metalinguagem seja o de veículo da metateoria. Já o uso da metalinguagem para dar o significado dos elementos da linguagem resolve apenas parte da questão da significação, pois para uma explicação completa do que é, e como se dá, a significação precisaríamos, agora, determinar qual o significado dos elementos da metalinguagem.

O que se impõe para uma explicação absoluta da significação é a exibição de um "mecanismo" que possa ele mesmo gerar a significação.

A tese sintaxista da significação afirma que as regras de manipulação de símbolos de um sistema formal são suficientes para gerar a significação. Vamos tratar desta tese sintaxista no âmbito de um exemplo, o sistema formal para a aritmética dos números naturais.

Conhecer o que significa ser um número natural poderia ser traduzido em conhecer os usos possíveis de numerais, para os números naturais, dentro de sentenças. Essencialmente esta tese é uma instância de uma tese mais geral que afirma que podemos determinar o significado de uma noção através do uso que fazemos desta noção, tese esta que eu considero respeitável, a princípio.

Se um sistema formal para os números naturais, no caso a aritmética elementar formalizada, apresenta todos os usos possíveis de numerais para números naturais, e se aceitamos que conhecer o significado de uma sentença é o mesmo que conhecer as condições de verdade desta sentença, e se admitimos que as condições de verdade para as sentenças da aritmética elementar dependem unicamente de saber se a sentença é demonstrável ou não, então poderíamos, a princípio, afirmar que o jogar pelas regras do sistema formal da aritmética elementar é suficiente para explicar o que

significa ser um número natural.

Tal tese, entretanto, não é sustentável, pois demonstração e verdade para os números naturais não são conceitos coextensivos. Os argumentos a serem apresentados na conclusão mostram o porque da não coextensividade.

F- O Sistema Formal Segundo Smullyan

O objetivo desta secção é o de definir exatamente o conceito de sistema formal usado no corpo desta tese, seguindo a definição de Smullyan ([11]).

Por um alfabeto K entende-se um conjunto ordenado finito de elementos chamados símbolos de K. Qualquer sequência linear finita, $x_1x_2...x_n$ de símbolos de K é chamada de palavra, ou expressão, ou string em K de tamanho n. Concatenar as palavras $x_1x_2...x_n$ e $y_1y_2...y_m$ de tamanhos n e m, respectivamente, significa obter a palavra $x_1x_2...x_ny_1y_2...y_m$ de tamanho $n+m$.

Definição I.1- Um sistema formal elementar (abreviado SFE) (E) é uma coleção dos seguintes itens:

1) O alfabeto K, que é um conjunto finito ordenado de símbolos;

2) Um conjunto de símbolos chamados variáveis;

3) Um conjunto de símbolos chamados predicados, a cada um dos quais é assinalado um inteiro positivo chamado grau do predicado;

4) Dois símbolos a mais chamados de símbolo de implicação ("-->") e símbolo de pontuação (","");

5) Uma sequência finita $A_1,...,A_n$ de sequências de símbolos que são fórmulas bem-formadas, chamados de axiomas do sistema.

6) Uma regra de substituição;

7) Uma regra de "destacamento" (semelhante ao MP).

Observação: os alfabetos de 1 a 4 devem ser mutuamente disjuntos.

Um termo t de (E) é qualquer sequência finita de símbolos que contenha variáveis e/ou elementos de K. Uma fórmula atômica de (E) é uma

expressão do tipo $Pt_1,\ldots,t_m$, onde P é um predicado de grau m, e $t_1,\ldots,t_m$ são termos. Uma fórmula F de (E) é dita bem-formada se F é atômica ou da forma $F_1 \rightarrow F_2 \rightarrow \ldots \rightarrow F_n$ onde cada F_i é atômico (lê-se $F_1 \rightarrow (F_2 \rightarrow (\ldots \rightarrow F_n))\ldots)$).

Uma palavra em K é uma sequência finita qualquer de símbolos de K, o conjunto de palavras em K denota-se $\underline{K}$. Uma instância de uma fórmula bem-formada X é uma sequência de símbolos obtida após a substituição das variáveis de X por palavras em K. Uma fórmula bem-formada que não contém variáveis é chamada de sentença.

Um teorema de (E) é qualquer sequência de símbolos que ou é um axioma de (E), ou é derivável dos axiomas de (E) por um número finito de aplicações das duas regras (6 e 7), ditas regras de inferência.

A regra de substituição permite a substituição das variáveis de uma fórmula bem-formada por quaisquer palavras em K.

A regra de destacamento, ou MP, permite inferir X_2, a partir de X_1 e $X_1 \rightarrow X_2$, desde que X_1 seja atômica.

Estabeleçamos uma convenção. Para representar um atributo P do seguinte tipo: Pa, Pxa, Pax, $Pxay$, onde x e y são variáveis, e onde a é um símbolo de K usamos o seguinte: $P\#a@$ onde $\#$ e $@$ são variáveis que podem ser substituídas pela palavra vazia, que usualmente representaremos por $\&$. Deste modo, onde encontramos um axioma do último tipo substituímos pelos quatro axiomas anteriores.

Pergunta: qual a razão de introduzir o conceito de SFE?

Um dos propósito é o de explicar a noção de definibilidade por recursão, ou seja, a definição recursiva de um conjunto qualquer. Em uma definição por recursão usamos: regras que indicam os elementos que pertencem ao conjunto (base), regras do tipo "se-então", e cláusula maximal. Em um SFE os axiomas correspondem aos axiomas de uma definição

recursiva, e a cláusula maximal corresponde, mais ou menos, as duas regras de inferência ([11] págs. 2 e 3). A cláusula maximal estabelece em geral que nenhum elemento pertence ao conjunto definido exceto aqueles que se seguem dos axiomas. No caso de um SFE substitui-se a cláusula maximal pela seguinte definição formal de "segue-se de": Um "string" X segue-se dos axiomas do SFE que define o conjunto se e somente se X pode ser obtido dos axiomas do SFE por um número finito de aplicações das duas regras de inferência.

Problema: o conjunto $\underline{K}$, de palavras sobre K, pode ser representado em um SFE (E) sobre K, contudo tal conjunto deve necessariamente ser dado na metalinguagem já que uma das regras de inferência (a substituição) necessita de tal conjunto para operar. Ou seja, se por um lado um SFE serve para explicar a noção de definibilidade por recursão por outro lado ele mesmo, o SFE, por definição, necessita de um conjunto que deve poder ser construido recursivamente. Colocando de outra forma, a definição de um SFE pressupõe dado na metalinguagem um conjunto recursivamente definível. Só que justamente um SFE tem o propósito de servir de ferramenta de definição recursiva de conjuntos. Melhor seria então se afirmássemos que o propósito de um SFE é o de representar uma definição recursiva de conjuntos.

Seja W um atributo em $\underline{K}$, isto é um subconjunto do conjunto das palavras em K. *Diz-se que W é formalmente representável sobre K se e somente se existe um predicado P tal que: X pertence a W se e somente se PX é provável em (E) de K* (X é aqui uma metavariável para elementos de $\underline{K}$). *Um atributo W é solúvel sobre $\underline{K}$ se e somente se W e o seu complemento $-W$ são formalmente representáveis sobre K.*

As definições de atributo recursivamente enumerável e atributo

recursivo são análogas as definições de formalmente representável e solúvel, respectivamente, com exceção que as mesmas são definidas para SFEs diádicos, onde a representação é binária (cardinal de $K = 2$).

Dado o conceito de SFE, estamos aptos a compreender o que afirma Smullyan no primeiro capitulo do seu livro "*Theory of Formal Systems*": "*The notions of formal system and mechanical operation are intimately connected; either can be defined in terms of the other.*" ([11] pág. 1). De fato, podemos definir uma máquina de processar simbolos como algo que é capaz de gerar um conjunto formalmente representável.

G- Referências Bibliográficas

[1] Carnap, R.; THE LOGICAL SYNTAX OF LANGUAGE, Routledge & Kegan Paul, 1959

[2] Carnap, R.; FOUNDATIONS OF LOGIC AND MATHEMATICS, em: International Encyclopedia of Unified Science, v.I, p. 139-213, The University of Chicago Press, 1955

[3] Carnap, R.; THE PHILOSOPHY OF RUDOLF CARNAP, em: The Library of Living Philosophers, v.XI, ed. P.A.Schilpp, Open Court, 1963

[4] Carnap, R.; INTRODUCTION TO SEMANTICS, Harvard University Press, 1975

[5] Davis, P.J. e Hersh, R.; THE MATHEMATICAL EXPERIENCE, Birkhäuser Boston, 1981

[6] Dummett, M.; THE PHILOSOPHICAL SIGNIFICANCE OF GODEL'S THEOREMS, em: Truth and Another Enigmas, p.186-201, Duckworth, 1978

[7] Hilbert, D.; ON THE FOUNDATIONS OF LOGIC AND ARITHMETIC, 1904 em: From Frege to Gödel, ed. J. Van Heijencort, p. 129-138, Harvard University Press, 1977

[8] Hilbert, D.; ON THE INFINITE, 1925 em: From Frege to Gödel, ed. J. Van Heijenoort, p. 367-392, Harvard University Press, 1977

[9] Hilbert, D.; THE FOUNDATIONS OF MATHEMATICS, 1927 em: From Frege to Gödel, ed. J. Van Heijenoort, p. 367-392, Harvard University Press, 1977

[10] Kleene, S.C.; INTRODUCTION TO METAMATHEMATICS, em:

Bibliotheca Mathematica, v.I, North-Holland, 1971

[11] Smullyan, R.M.; THEORY OF FORMAL SYSTEMS, em: Annals of Mathematics Studies, 41, Princeton University Press, 1961

CAPITULO III

MAQUINAS DE SISTEMAS FORMAIS

A- Introdução

Eu terminei o capítulo anterior definindo uma máquina capaz de processar símbolos como algo que é capaz de gerar os elementos de um conjunto formalmente representável.

Os sistemas formais, segundo a caracterização da última secção do capítulo anterior, servem basicamente para descrever, de maneira finita, conjuntos infinitos. 'Usualmente diz-se que uma máquina, que trabalha com símbolos, também define um conjunto (possivelmente infinito), de modo que sendo finita a descrição de tal máquina o seu papel se assemelha ao do sistema formal. Contudo esta é uma semelhança apenas aparente. Neste capítulo eu pretendo discutir esta questão.

Definir o que é uma máquina, passo fundamental para criar uma ciência da máquina ("maquinologia"), é um problema difícil de resolver, o que não nos impede de tentar dar uma caracterização ainda que parcial. As máquinas são objetos construídos fisicamente pelo homem, construídas para realizar determinadas operações. Esta caracterização, que está longe de ser uma definição, nos permite distinguir as máquinas de quaisquer outros objetos naturais, ou seja que não receberam trabalho humano, e de outros objetos que receberam trabalho mas que não realizam operações, como uma obra de arte, um edifício, ou um fio de algodão, por exemplo.

De todo o modo, as máquinas que nos interessam aqui são aquelas capazes de processar símbolos.

Este capítulo começa com uma discussão puramente teórico-matemática acerca das máquinas sob a designação genérica de autômatos. Em segundo lugar serão apresentadas, ainda que de um ponto de vista teórico, as máquinas computadoras digitais programáveis. Por fim, serão traçadas algumas considerações inter-relacionando estes tres primeiros capítulos.

O objetivo deste capítulo é o de mostrar que os computadores digitais são "máquinas de sistemas formais".

B- Autômatos

Sob o título "autômato" podemos entender todas aquelas máquinas capazes de operar de maneira espontânea. Particularmente vamos usar a designação autômato para nos referir a máquinas tais que a sua operação física possa ser descrita como uma operação sobre símbolos. Isto não significa que vamos examinar, neste momento, o funcionamento de máquinas como objetos físicos. A idéia é utilizar descrições teórico-matemáticas, onde fazemos abstrações dos elementos estruturais de uma máquina e do seu funcionamento. A classe de tais descrições definem por extensão o conceito de "autômato". Assim, segundo o sentido que eu utilizo aqui, autômato é um conceito matemático que descreve as propriedades estruturais de uma certa classe de máquinas.

Matematicamente falando, sob o título "autômato", entendido em um sentido genérico, coloca-se o estudo de processos efetivos, seja para a descrição de conjuntos infinitos, seja para a execução de funções. Os autômatos devem, basicamente, caracterizar o que é, ou possa ser, um processo efetivo.

Diz-se que um processo é efetivo se ele é executável por uma máquina de Turing tal que existe um tempo t em que a máquina para.

O estudo de autômatos se extende a diversos tipos abstratos de máquinas, e estes diversos tipos abstratos de máquinas refletem diferentes tipos de estruturas e variações de estruturas.

Eu entendo que o estudo de autômatos tem caráter a priori. Tal afirmação é controversa. O que significa ter caráter a priori? Significa que dada a caracterização de um autômato é possível demonstrar como

teorema todas as propriedades deste autômato. Colocado de outro modo, considerações de caráter a posteriori não são levadas em conta para na caracterização de um autômato, ou na caracterização do que é um processo efetivo.

Ilustrando: sabemos que o nosso sistema solar não durará mais que um número finito de anos. Esta afirmação, que pertence a física, tem caráter a posteriori. Entretanto, ainda que tal afirmação implique a impossibilidade da existência de um autômato localizado em algum lugar do sistema solar que faça uma computação com um tempo maior que a duração do sistema solar, a afirmação não aporta nada ao estudo de autômatos. Justamente os fatores que são levados em consideração no estudo dos autômatos são as características estruturais e formais de caráter a priori que podem influenciar o "funcionamento" do autômato. E' perfeitamente concebível um autômato que realize uma computação com duração maior que a do nosso sistema solar.

Ninguém jamais construiu um autômato e passou a observá-lo para descobrir através do comportamento do mesmo quais eram as suas propriedades, mesmo porque neste caso este alguém poderia no máximo formular leis de regularidade, que por seu turno são suscetíveis apenas de confirmação parcial. As máquinas das quais tratamos, se me é permitido assim falar, estão mais no "mundo das idéias do que no mundo real", pois o estudo dos processos efetivos tem caráter a priori.

O objetivo desta secção é o de mostrar a conexão entre o conceito de sistema formal e o conceito de autômato. Antes é necessário caracterizar melhor o que eu estou chamando de autômato. A literatura além de diversificada não fornece uma concepção unificada a este respeito. Deste modo a caracterização, a seguir, é feita tendo em vista os propósitos específicos desta dissertação.

Um autômato descreve uma máquina discreta, logo, uma que trabalha com tempo discreto, portanto enumerável, sendo que suas operações são invariantes no tempo, ou seja a sua ação não se altera em tempos distintos.

As tres funções executadas por um autômato são: reconhecer, gerar e computar; sendo que reconhecer e gerar podem ser vistas como casos particulares de computar. Eu destaco a função gerar, pois considero que o seu tratamento envolve algumas particularidades marcantes, principalmente porque, no capitulo II, tínhamos chegado a conclusão que poderíamos definir uma máquina como algo capaz de gerar elementos de um conjunto formalmente representável. Em geral diz-se que um autômato é um reconhecedor ("acceptor") se ele executa a função de reconhecer; que ele é um transdutor ("transducer") se ele executa a função de computar; gerador ("generator") se ele executa a função de gerar.

O gerador, para nossos propósitos, pode ser definido como um transdutor que recebe um número natural de algum modo codificado e gera uma palavra que possui ordem igual àquela do número natural fornecido, ou, alternativamente, pode ser definido como um transdutor que recebe uma sequência de escolha como entrada. Uma peculiaridade do gerador é que ele tem de ser capaz de gerar todas as palavras do conjunto formalmente representável.

Definição II.1- Um autômato é uma quintupla $\langle Q, X, Y, F, \emptyset \rangle$ onde Q é o conjunto de estados do autômato, X é o alfabeto de entrada e Y é o alfabeto de saída, F é uma função de $Q \times X \rightarrow Q$, dita função de transição de estados, e $\emptyset$ é uma função de $Q \times X \rightarrow Y$ dita função de saída.

Podemos ver as intruções de um autômato como uma quádrupla $\langle q, x, F(q,x), \emptyset(q,x) \rangle$.

Cabe aqui uma pequena observação. Geralmente tratamos o conjunto de estados como um alfabeto, onde cada elemento do alfabeto designa um estado diferente. Este artifício parece ser bastante útil.

Definição II.2- Um autômato inicializado é um par $\langle A, q_0 \rangle$, onde A é um autômato, segundo a definição II.1, e q_0 é um estado destacado, que pertence a Q_A.

Então, considerando-se um autômato A e um estado destacado, dito estado inicial, q_0:

$$q(t+1)=\mathcal{Y}(q(t),x(t))$$

$$y(t)=\mathcal{O}(q(t),x(t))$$

onde t é um tempo discreto; $q(t)$, $q(t+1)$ pertencem a Q_A, $x(t)$ pertence a X_A e é o resultado de dar entrada em um símbolo no tempo t (comumente descrito como leitura de um símbolo e deslocamento da cabeça de leitura para o próximo símbolo); $y(t)$ pertence a Y_A e é dito o resultado da saída no tempo t. Deste modo $q(1)=q_0$ e então o autômato define uma função, ou operador sobre palavras, $T(A,q_0)$ tal que, para $x=x(1)x(2)...x(r)$, $T(A,q_0)(x)=y=y(1)y(2)...y(r)$.

Volto a ressaltar, a definição acima fornecida é bastante genérica. Há casos em que o autômato recebe a entrada de várias fitas, ou tem fitas de trabalho do tipo pilha, etc. Também é possível impor restrições aos vários componentes da quíntupla que define um autômato resultando assim diferentes tipos de autômatos. Há ainda os casos degenerados como o dos autômatos sem memória, autômatos autônomos etc. O autômato sem saída é um caso interessante. Tal autômato pode ser descrito por uma tripla $\langle Q, X, Y \rangle$. O comportamento de tal máquina não pode ser descrito em termos de operadores sobre palavras. Mais interessante ainda é o autômato ancorado. Ele é descrito por uma tripla $\langle A, q_0, Q' \rangle$ onde A é um autômato sem saída, q_0 é o estado inicial, e portanto este autômato é inicializado, e Q' está

contido em Q_A. O conjunto de palavras de entrada (possivelmente vazio), que leva a máquina do estado inicial q_0 até um estado q_i pertencente a Q', é dito o conjunto de palavras reconhecível pelo autômato.

Assim, um reconhecedor é um autômato ancorado segundo a caracterização anterior.

Vamos, agora, ao ponto que interessa investigar nesta secção, o paralelo entre o conceito de sistema formal e o conceito de autômato. Em primeiro lugar, eu quero mostar que um autômato realiza um sistema formal de uma maneira a ser precisamente definida. Em segundo lugar eu quero mostrar que um autômato pode ser descrito por um sistema formal. A palavra "realiza" foi escolhida para indicar que há uma ação.

O que pretende o autor desta dissertação com a palavra realizar? Minha intenção é mostrar que, basicamente, um autômato transforma símbolos segundo regras pré-determinadas. Ora, esta capacidade de seguir regras é a chave para entender o uso da palavra realizar. Se um sistema formal define um ou vários conjuntos formalmente representáveis através de regras podemos dizer de qualquer máquina apta a seguir estas regras que ela realiza o sistema formal. Então, de um certo modo, um lógico realiza um sistema formal porque ele é capaz de executar as regras deste sistema formal para, por exemplo, provar um teorema.

Uma função computável é uma relação tal que para um subconjunto do domínio dado um elemento pertencente a este subconjunto podemos exatamente determinar qual o elemento correspondente, pertencente a imagem da função. Acredito que para toda regra sintática podemos associar uma função. Sendo assim, podemos ver cada regra sintática de um sistema formal, seja de formação ou de transformação, como uma função. Se realmente podemos ver toda regra sintática como uma função isto poderia

nos fornecer uma explicação de como uma máquina pode seguir uma regra. Como podemos organizar uma máquina tal que a transição de estados e a saída podem ser vistas como funções, então podemos adequadamente organizar a transição entre estados e a saída de uma máquina de tal forma que a função transição e a função saída sejam isomorfas a uma função que representa uma regra sintática.

A função de transição pode ser vista como uma regra que determina qual o próximo estado da máquina, dado que no presente momento ela se encontra em um estado *q* e recebeu o símbolo *x* como entrada. Semelhantemente a função de saída também pode ser vista como uma regra que determina a saída da máquina, dado que no presente momento ela se encontra em um estado *q* e recebeu o símbolo *x* como entrada.

Uma máquina realiza operações para obter um determinado resultado. Para alcançar tal resultado ela precisa poder determinar que operações realizar. Uma maneira bastante adequada de determinar que operação uma máquina deve realizar é fazer a operação depender de um estado da máquina imediatamente anterior a operação tanto quanto de um dado vindo do exterior desta máquina, e que portanto não pode fazer parte do estado da mesma. Deste modo, utilizar os estados da máquina como controle das operações parece ser um modo adequado de realizar um controle pensado a partir de regras. Como a transições entre estados podem ser arranjadas como funções, e como regras sintáticas podem ser vistas como funções, acredito, então uma máquina pode operar a partir de regras.

Resumindo, uma regra sintática descreve uma função, segundo eu penso. Os autômatos são máquinas capazes de seguir regras adequadamente codificadas e criteriosamente distinguidas estruturalmente pelos seus estados internos (estados de controle). Os sistemas formais são basicamente um conjunto de regras sintáticas que descrevem um ou mais

conjuntos infinitos, conjuntos estes ditos formalmente representáveis. Os autômatos com relação aos conjuntos formalmente representáveis podem realizar tres tarefas, reconhecê-los, gerá-los e processá-los.

Cabe aqui um pequeno "insight". Nós, seres humanos, somos capazes (alguns ao menos) de compreender a descrição de um conjunto infinito através de uma descrição finita dada por um sistema formal. De que maneira? Aplicando as regras estabelecidas pelo sistema. Uma máquina também é capaz, de algum modo de aplicar estas regras. Porém quando a partir da descrição de um autômato nós, enquanto seres humanos, processamos os dados que o autômato processa, ou processou, não estamos realizando um autômato, mas estaremos simulando um autômato. A grande diferença existente entre um autômato e um sistema formal é que o primeiro possui comportamento enquanto o segundo não. Melhor dizendo, o autômato enquanto conceito não pode possuir comportamento, mas os objetos aos quais os autômatos caracterizam podem possuir comportamento. Só um comportamento pode ser simulado. Justamente porque um autômato é temporal, enquanto um sistema formal é atemporal - está fora do tempo -, é que ele pode apresentar comportamento. A variável tempo não desempenha nenhum papel para um sistema formal!

Proposição II.1- Dizemos que um autômato *A* realiza fracamente um sistema formal *S* quando o conjunto reconhecido, gerado, enfim processado pelo autômato *A* é igual ao conjunto formalmente representável pelo sistema formal *S*.

Proposição II.2- Dizemos que um autômato *A* realiza um sistema formal *S* se ele realiza fracamente o sistema formal *S* e existe uma função de tradução das regras do sistema formal *S* na função de transição e de saida do autômato *A* e vice-versa.

Resumindo, um autômato não é um sistema formal, mas um autômato pode ser descrito por meio de um sistema formal, além de ele mesmo realizar um sistema formal.

Vamos mostrar agora como é possível descrever um autômato por meio de um sistema formal.

Proposição II.3- Seja *A* um reconhecedor (acceptor) genérico tal que:

$Q=\{q_0,\ldots,q_n\}$ é o alfabeto de estados, e

$X=\{a_1,\ldots,a_m\}$ é o alfabeto de entrada e os dois são disjuntos. Um SFE *(E)* que descreve este autômato deve ser construído da seguinte forma:

K= Q união X

e os axiomas são: (observação: onde aparece a palavra grupo significa que deve existir um grupo de axiomas que respeitem a forma genérica fornecida, caso contrário o axioma é único)

Grupo 1) Xa_1 ; ... ; Xa_m onde *X* é um atributo que representa todos os símbolos de entrada

Grupo 2) *Xx-->Ex* ; *Ex-->Ey-->Exy* onde *E* é um atributo que representa todas as sequências possíveis de entrada

Grupo 3) Fq_1 ; ... ; Fq_j onde *F* é um atributo que representa todos os estados finais

4) *Ix-->Ex-->D1,x,qo* onde *I* é um atributo que representa o "input" para o autômato (deve portanto ser instanciado antes do funcionamento), e *D* representa a descrição do autômato em um instante de tempo *t*, neste caso a descrição é inicializada no tempo *1* com a entrada dada pelo input *I* e no estado inicial *qo*

Grupo 5) *Dx,y@,z-->Xy-->Dx1,@,w* ; ... que representam as regras de operação do autômato em que ele recebe um certo símbolo de entrada *y* em um instante *t* e passa do estado *z* para o estado *w* no tempo *t+1*

6) *Dx,&,z-->Fz-->Ix-->Ax* onde *A* é o atributo que representa a

palavra reconhecida quando a entrada já estiver vazia (&) e o estado z resultante pertencer ao conjunto dos estados finais.

Observe-se que para descrever o autômato por meio de um SFE eu tive que utilizar um artifício para representar o tempo dentro da descrição do autômato, pois um autômato depende, para o seu funcionamento, da variável tempo.

Proposição II.4- Seja T um transdutor ("transducer") genérico tal que:

$Q=\{q_0,\ldots,q_n\}$ é o alfabeto de estados, e

$X=\{a_1,\ldots,a_m\}$ é o alfabeto de entrada, e

$Y=\{b_1,\ldots,b_p\}$ é o alfabeto de saída, sendo que qualquer um dos dois alfabetos de símbolos, seja o de entrada seja o de saída, é mutuamente disjunto do alfabeto de estados. Um SFE (E) que descreve este autômato deve ser construído da seguinte forma:

$K=Q$ *união* X *união* Y

e os axiomas são:

Grupo 1) Xa_1 ; ... ; Xa_m onde X é um atributo que representa todos os símbolos de entrada

Grupo 2) $Xx\text{-->}Ex$; $Ex\text{-->}Ey\text{-->}Exy$ onde E é um atributo que representa todas as sequências de entrada possíveis

Grupo 3) Fq_1 ; ... ; Fq_j onde F é um atributo que representa todos os estados finais (pode existir ou não)

4) $Ix\text{-->}Ex\text{-->}D1,x,\&,q_0$ onde I é um atributo que representa o "input" para o autômato (deve portanto ser instanciado antes do funcionamento), D representa a descrição do autômato em um instante de tempo t, neste caso a descrição é inicializada no tempo 1 com a entrada dada pelo input I, a palavra de saída vazia (&) e no estado inicial q_0

Grupo 5) $Dx,y@,z,w\text{-->}Xy\text{-->}Dx1,@,zb_1,s$; ... que representam as regras de operação do autômato em que ele recebe um certo símbolo de entrada y em um instante t e passa do estado w para o estado s no tempo $t+1$ justaponto a palavra de saída z um símbolo de saída b_1 pertencente a Y

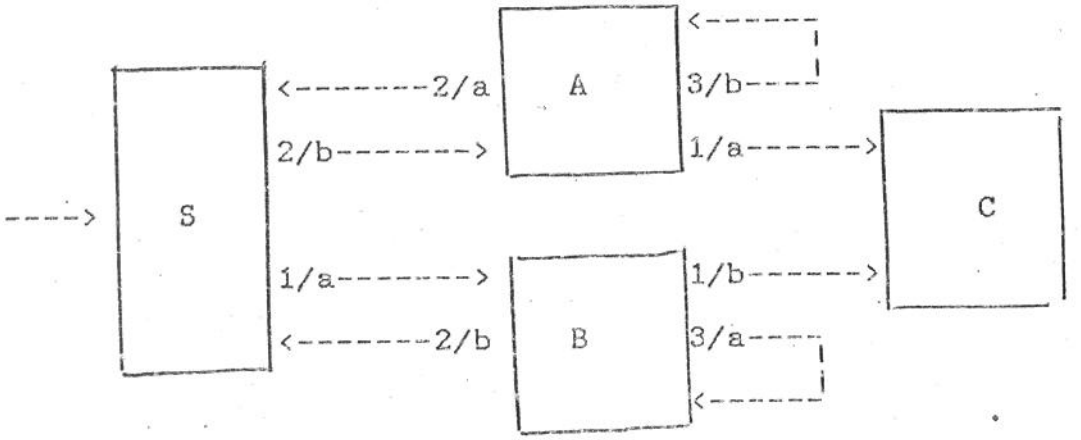
6) $Dx,\&,z,w\text{-->}Tw,z$ onde T é o atributo que representa a palavra de saída z no estado final w quando a entrada estiver vazia ($\&$) (podemos intercar a fórmula Fw na fórmula anterior quando houver conjunto de estados finais).

Os transdutores também são chamados de máquinas sequenciais, pois para cada palavra recebida ele gera uma palavra de saída.

Como os geradores podem ser vistos como transdutores que recebem como entrada uma sequência que possibilita fazer escolhas com relação as palavras a serem geradas, então vale a proposição II.4 para os geradores. Uma palavra é dita gerável, por um gerador, se e somente se existe uma ou várias upla(s) de palavra(s) de entrada de modo tal que o transdutor produza a palavra de saída para tal entrada.

Vamos ver como funciona tal descrição em um exemplo concreto de um gerador para a seguinte gramática $G=(V_t,V_n,S,P)$ onde:

$V_t=\{a,b\}$; $V_n=\{S,A,B,C\}$; $P=\{S\text{->}aB;\ S\text{->}bA;\ A\text{->}a;\ A\text{->}aS;\ A\text{->}bA;\ B\text{->}b;\ B\text{->}bS;\ B\text{->}aB\}$ e F é o estado final, um grafo para tal autômato é:



Neste autômato gerador (na verdade um transdutor) o primeiro símbolo indica o símbolo lido e o segundo símbolo o símbolo resultante. Uma palavra é gerável se e somente se existe uma sequência composta de *1's*, *2's* e *3's* tal que a máquina produza a palavra desejada.

Vamos representar este gerador em um SFE. Seja um SFE *(E)* com *K={ 1,2,3,a,b,S,A,B,C}* , onde os axiomas são os seguintes:

1) *X1* 2) *X2* 3) *Xx-->Ex* 4) *Ex-->Ey-->Exy* tal que *E* significa entradas admissíveis

5) *FC* *C* é um estado final

6) *Ix-->Ex-->D1,x,&,S* tal que se a máquina recebe um "input" *x* e se este "input" é admissível então o descritor no tempo *1* contém o "input" *x*, a palavra de saída vazia e o símbolo *S* que neste caso representa o estado inicial

7) *Dx,1@,z,S-->Dx1,@,za,B*

8) *Dx,2@,z,S-->Dx1,@,zb,A*

8) *Dx,1@,z,A-->Dx1,@,za,C*

10) *Dx,2@,z,A-->Dx1,@,za,S*

11) *Dx,3@,z,A-->Dx1,@,zb,A*

12) *Dx,1@,z,B-->Dx1,@,zb,C*

13) *Dx,2@,z,B-->Dx1,@,zb,S*

14) *Dx,3@,z,B-->Dx1,@,za,B*

todas estas são regras de produção de modo tal que a aplicação da regra ao descritor "avança" o autômato uma unidade de tempo

15) *Dx,&,z,w-->Fw-->Tw,z* tal que se o descritor tem a palavra vazia *&* no campo de entrada então a

palavra que foi gerada pertence ao conjunto T de palavras geradas se o estado é um estado final

Obs: o símbolo @ é substituível por qualquer palavra, a vazia inclusive, conforme convenção do capítulo I.

Neste exemplo eu apresento a descrição de um gerador que por sua vez realiza uma gramática regular. No caso de outras gramáticas a descrição deve sofrer algumas alterações de modo a respeitar as peculiaridades de cada gramática.

C- Computadores

Deste ponto em diante eu passo a chamar um computador digital de máquina de sistemas formais. Com isso ficam bem separadas e caracterizadas a ação, descrita por um sistema formal, do agente, a máquina bruta.

O objetivo desta secção é o de caracterizar brevemente um computador, mas somente aqueles caracteres que venham a ser relevantes teóricamente, portanto sem considerar peculiaridades de cada máquina fisica.

Definição II.3- Uma máquina de sistemas formais M consiste da especificação dos seguintes conjuntos e funções:

a) Um conjunto E chamado conjunto de entrada;

b) Um conjunto V chamado conjunto de memória;

c) Um conjunto S chamado conjunto de saida;

d) Uma função $E_M: E \to V$ chamda de função de entrada;

e) Uma função $S_M: V \to S$ chamada de função de saida;

f) Um conjunto $O_M=\{o_1,o_2,\ldots\}$ onde, para $i>0$, $o_i: V \to V$, chamado de conjunto de operações;

g) Um conjunto $T_M=\{t_1,t_2,\ldots\}$ onde, para $i>0$, $t_i: V \to \{v,f\}$, chamado de conjunto de testes.

Obs: as funções de d) a g) são funções totais.

Uma máquina deve suprir o significado dos testes e operações presentes em um programa. Cada identificador de operação denota uma transformação na estrutura de memória da máquina, e por sua vez cada identificador de teste denota uma função de verdade. As funções de

entrada e saída permitem à máquina manipular informação de e para a estrutura de memória. Portanto a caracterização dada acima é de uma máquina que permite a interpretação de um programa, de tal maneira que dada a definição do que é um programa, podemos definir o que é uma computação.

Duas características importante dos computadores digitais é que eles possuem um controle interno e uma memória principal. Tal memória é constituída por um conjunto de registros cada um dos quais é capaz de armazenar um valor arbitrário de um conjunto de valores previamente determinado. Em adição, as operações e testes de tais máquinas são definidos apenas sobre conteúdos de registros específicos ao invés de sobre toda a memória. Usualmente designamos tais máquinas como máquinas de registros.

Definição II.4- Uma máquina de registros, Mr é uma máquina de sistemas formais M onde:

a) $V=A^n$, para algum conjunto A, em uma máquina de n registros;

b) para o_i pertencente a O_{Mr} e R_j j-ésimo registro a operação tem a seguinte forma $R_j:=o_i(R_j)$, onde ":=" indica que o valor resultante da expressão a direita passa como conteúdo para o registro designado pelo nome a esquerda;

c) para t_i pertencente a T_{Mr} e R_j j-ésimo registro o teste tem a seguinte forma $t_i(R_j)$.

Passemos a chamar uma máquina de registros M_r que possui controle interno de máquina objetivo. Sobre tal máquina, que é a descrição completa de um moderno computador digital, são interpretados os programas.

Definição II.6- M_{PH} é chamada de uma máquina virtual, e ela é o resultado de interpretar um artefato de software sobre uma máquina-

objetivo *H*.

Proposição II.5- Para toda máquina virtual M_{PH} existe um autômato *A* que a descreve.

D- Retrospectiva

Haugeland, no seu livro, apresenta os computadores como sistemas formais automáticos ([4] pág 48). A meu ver tal caracterização é incorreta, pois a designação "automático" não cabe de modo nenhum ao conceito de sistema formal. Como vimos um sistema formal não pode apresentar comportamento.

O conceito de sistema físico de símbolos de Newell e Simon deve, a meu ver, estar intimamente relacionado com o conceito de máquina de sistemas formais. Porém, não podemos afirmar a equivalência dos dois sem nos envolver em uma polêmica. Para Newell e Simon a máquina de Turing, ou máquina universal, é apenas o primeiro passo na direção ao sistema físico de símbolos ([8] pág. 117). Neste conceito de máquina universal ainda não estão presentes as noções completas de interpretação e designação, segundo estes autores. São ainda necessárias duas qualidades, ou capacidades, adicionais: primeiro, a capacidade de armazenar um programa e tratá-lo como se fosse um dado ([8] pág. 117); segundo, a capacidade de processamento de listas ([8] pág. 118). A primeira completa a capacidade de interpretação, e a segunda fornece a capacidade de designação, noções centrais ao conceito de sistema físico de símbolos segundo os autores ([8] pág. 116).

Contudo, tais capacidades, de tratar os programas como dados e de processar listas, ainda que façam diferença do ponto de vista da atividade de programação e do uso dos computadores não modificam a classe das funções que uma máquina pode computar. A meu ver, um sistema físico de símbolos é essencialmente uma máquina de sistemas formais.

O conceito de Newell e Simon é bastante claro quando realça a palavra "físico" denotando com isso um artefato obtido por engenharia e que obedece leis da física. A palavra "máquina" presente na descrição "máquina de sistemas formais" tem um papel bastante semelhante. Com ela eu pretendo apontar o agente que deve ser material e que deve obedecer leis da física. As palavras "sistemas formais" estão a indicar em que consiste a acção que a máquina pode efetuar. Segundo o sentido dado aqui ao conceito de sistema formal, de ser algo representável ou se preferirem "escrevível", a máquina pode adequadamente ser pensada como capaz de realizar o sistema formal justamente porque nós traçamos uma interconexão entre os estados da máquina e os símbolos e regras para manipulação de símbolos.

Há então dois aspectos neste conceito de máquina de sistemas formais: o aspecto físico, marcado pela palavra máquina; e o aspecto imaterial da ação marcado pelas palavras "sistemas formais".

D- Referências Bibliográficas

[1] Beckman, F.S.; MATHEMATICAL FOUNDATIONS OF PROGRAMMING, Addison-Wesley, 1981

[2] Bird, R.; PROGRAMS AND MACHINES, John Wiley & Sons, 1976

[3] Haeberer, A.M. e Veloso, P.A.S.; WHY SOFTWARE DEVELOPMENT IS INHERENTLY NON-MONOTONIC: A FORMAL JUSTIFICATION, PUC-RJ, 1989 não publicado

[4] Haugeland, J.; ARTIFICIAL INTELLIGENCE: THE VERY IDEA, MIT Press, 1987

[5] Hopcroft, J.E. e Ulmman, J.D.; FORMAL LANGUAGES AND THEIR RELATION TO AUTOMATA, Addison-Wesley, 1969

[6] Hopcroft, J.E. e Ulmman, J.D.; INTRODUCTION TO AUTOMATA THEORY, LANGUAGES, AND COMPUTATION, Addison-Wesley, 1979

[7] Nelson, R.J.; INTRODUCTION TO AUTOMATA, John Wiley & Sons, 1968

[8] Newell, A. e Simon, H.A.; COMPUTER SCIENCE AS EMPIRICAL ENQUIRY: SYMBOLS AND SEARCH, Comm. of ACM, v. 19, nro. 3, março/1976

[9] Scott, D.; SOME DEFINITIONAL SUGGESTIONS FOR AUTOMATA THEORY, Journal of Computer and System Sciences: 1, 187-212, 1967

[10] Trakhtenbrot, B.A. e Barzdin, Ya.M.; FINITE AUTOMATA, North-Holland, 1973

CAPITULO IV

PROGRAMAS E TEORIAS FORMAIS

A- Introdução

Parece ser geralmente aceito que os computadores são máquinas de sistemas formais. A exposição dos capítulos anteriores apenas aponta e organiza os conceitos envolvidos em tal definição. Tal definição revela apenas que o funcionamento de uma máquina virtual deve poder ser descrito por um sistema formal, onde justamente o programa é a regra para o funcionamento. Contudo, creio que um passo a frente deve e pode ser dado na compreensão do papel do conceito de sistema formal dentro da teoria da computação. Como foi dito anteriormente, um sistema formal é uma maneira adequada de expressar uma teoria formalmente. Nesta formalização aparecem em relevo as propriedades básicas de uma certa teoria, seus axiomas, e o aparato lógico-dedutivo capaz de permitir a extração de consequências desta teoria, seja para realizar previsões, seja para realizar confirmações.

O propósito deste capítulo é o de levar mais adiante a compreensão do papel do conceito de sistema formal com relação a teoria da computação.

Para começar vamos verificar porque um programa é uma regra. Em segundo lugar vamos tratar da questão que, epistemologicamente falando, apresenta maior importância. Trata-se de compreender a relação que existe entre um artefato de software e o conceito de teoria, mais especificamente o conceito de teoria formal, ou formalizável. Para tal

creio ser apropriado examinar a atividade de engenharia de software, onde estão envolvidos, na maioria das vezes, um engenheiro de software junto com um especialista buscando montar um artefato de software para determinados fins. Tal trabalho apresenta algumas semelhanças com o trabalho cientifico. De fato a atividade de engenharia de software pode ser comparada muito proximamente a uma atividade cientifica. Nesta última secção a palavra "problema" será bastante utilizada, e para uma caracterização do significado pretendido desta palavra aconselho como referências o livro de Haeberer et alli, "Formalizacion del Processo de Desarrollo de Software" ([5]) e o artigo de Veloso, "Outlines of a Mathematical Theory of General Problems" ([9]) constantes da bibliografia.

B- Programas como Regras

Uma linguagem de programação é definida por meio de uma gramática formal. Toda gramática formal, por sua vez, é um sistema formal, segundo a caracterização do segundo capítulo. Logo, toda linguagem de programação é definida por intermédio de um sistema formal.

Entretanto, o sistema formal que define uma linguagem de programação não é propriamente um cálculo. Na verdade, uma gramática formal define unicamente quais são as expressões pertencentes a linguagem que ela define. Uma gramática formal define um conjunto de expressões bem-formadas, logo só possui regras de formação. Para ser um cálculo seriam necessárias ainda as regras de transformação.

No caso especifico das linguagens de programação, as expressões da linguagem são textos de programa. Como a gramática de uma linguagem de programação é formal, todo texto de programa é uma expressão formal.

Um texto de programa, enquanto expressão formal, pode ser visto como um conjunto de sentenças. Segundo o tipo da linguagem de programação, nós podemos ter: textos imperativos, onde as sentenças do texto são imperativas, exemplo: Pascal; e textos assertivos, onde as sentenças do texto são asserções, exemplo: Prolog. No primeiro caso as sentenças do texto não podem ser ditas verdadeiras ou falsas, pois uma sentença imperativa é uma ordem. No segundo caso sim. Basicamente as teorias são um conjunto de asserções sobre um determinado objeto, tal que estas asserções podem ser ditas verdadeiras ou falsas. Portanto um programa assertivo pode expressar adequadamente uma teoria.

No caso das linguagens imperativas, o programa, como expressão

formal, denota uma função: a função executada sobre os dados de entrada segundo o algoritmo descrito pelo programa. Como um algoritmo é uma regra, todo programa imperativo é uma regra, no caso uma regra sintática.

Se um programa imperativo algoritmico é uma regra sintática podemos compreender como uma máquina de sistemas formais pode realizar um sistema formal. Simplesmente porque ela realiza um sistema formal através da execução de uma função descrita por uma regra, a saber, o texto de programa imperativo. Mais que isso, o sistema formal é definido por meio de regras, logo parece natural concluir que um programa que define um algoritmo define uma parte importante do sistema formal realizado pela máquina: a(s) regra(s) de transformação. As condições de entrada e saída delimitam os estados iniciais e os estados finais de uma máquina virtual. Tais estados estão para o programa como as expressões para a regra sob as quais opera. Logo o conjunto de expressões sobre as quais o programa (ou a regra) opera é o conjunto de estados da máquina, ou mais restritamente o conjunto de estados correspondente as possíveis combinações da parte da memória para os dados, com os quais o programa trabalha.

E´ possível representar, no caso de textos assertivos (um programa Prolog, por exemplo) dois tipos de sentenças: particulares, onde os termos da sentença são constantes descritivas; e regras que contém quantificadores. Um texto assertivo expressa uma teoria se nele existe alguma regra formulada, pois naturalmente um amontoado finito de sentenças particulares dificilmente poderia ser uma teoria.

A possibilidade de representar regras e sentenças particulares dentro de um texto assertivo, somada a capacidade que o interpretador do texto assertivo tem de funcionar segundo regras de transformação, no caso particular do Prolog através de resolução, ou seja por redução ao absurdo, produzem um cálculo, de tal modo que a máquina Prolog pode

extrair consequências lógicas a partir de um texto que contenha sentenças particulares e/ou regras.

C- A Relação Entre Programa e Teoria

Considerada na sua diversidade, a atividade de engenharia de software parece, a primeira vista, fugir a qualquer esquema. Isto é, observando as mais variadas pessoas programando os mais variados sistemas, a primeira vista, podemos pensar que há muito poucas coisas em comum nestas diversas atividades.

O texto que segue busca exatamente mostrar o que há de comum em toda atividade de engenharia de software. Nele eu desejo apresentar um modelo que capture as tarefas que obrigatoriamente estão envolvidos na referida atividade. E´ bem verdade que, talvez, não possamos em casos concretos verificar facilmente as correspondências pretendidas. E´ possível que algumas tarefas descritas não sejam explicitamente observáveis na diversidade a que eu me referi anteriormente, mas acredito que estas tarefas devem, ainda que implicitamente, estar presentes em qualquer processo de desenvolvimento de software. Segundo penso, estas tarefas são: teorização acercade um objeto; e formalização desta teorização, ainda que parcial.

A argumentação da conclusão depende, em parte, da existência de tarefas canônicas, ainda que não explicitamente visualizáveis.

O modelo a ser apresentado pretende ser uma descrição e não uma prescrição ao processo de engenharia de software. O pressuposto fundamental para que esta descrição seja possível é o pressuposto que afirma ser a atividade de engenharia de software uma atividade racional e construtiva, que inicia com a intenção de resolver um problema.

Podemos naturalmente dividir o processo de desenvolvimento de

software em dois momentos: o primeiro momento é o que começa com a intenção de produzir um software para determinados fins até a realização de uma especificação formal do que deve realizar este software; o segundo começa a partir desta especificação até alcançar o sistema devidamente programado. De maneira genérica poderíamos dizer que a primeira fase é a fase de trabalho do analista com o usuário ou "expert" que deseja o artefato de software. A segunda fase é aquela onde o programador e o analista estão envolvidos, neste caso o analista funcionando mais como um coordenador de tarefas. Tal distinção não implica que estas fases não tenham momentos concomitantes.

A exposição que eu apresento de aqui por diante nesta secção está fortemente calcada nas idéias apresentadas por Turski e Maibaum no livro "The Specification of Computer Programs" ([8]) e nas idéias apresentadas por Veloso, Haeberer e Baum no livro "Formalización del Processo de Desarrollo de Software" ([5]). Quando for o caso, os meus comentários pessoais estarão devidamente sinalizados.

Um domínio de aplicação é uma situação, um estado de coisas, sob exame intencional visando a produção de um artefato de software para realizar alguma tarefa dentro deste domínio. Como em qualquer domínio de investigação científica, um determinado ponto de vista orienta quais os predicados, relações e objetos relevantes para os propósitos em questão.

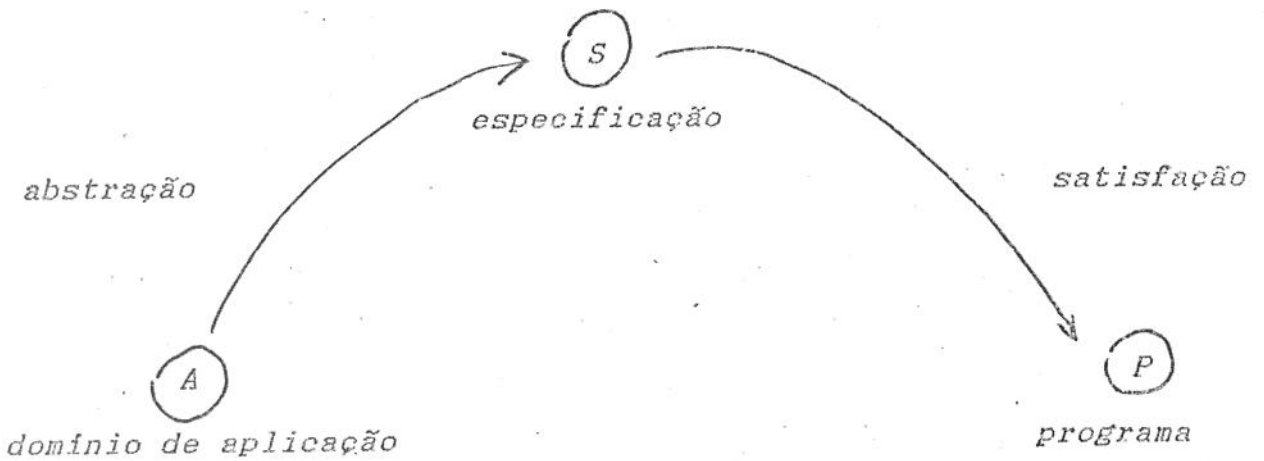
Uma especificação, por sua vez, é uma abstração, algo que representa em termos abstratos a visão que se tem do domínio de aplicação. A especificação é um objeto formal, constituído por um conjunto de sentenças, ou que se reduz a um conjunto de sentenças, que estabelecem relações relevantes e primitivas bem como restrições a estas relações. Em outras palavras uma especificação é o que poderíamos chamar de uma teoria. A especificação, por outro lado, descreve o que um programa deve

realizar para que ele, o programa, sirva aos propósitos que alguém tem em mente com respeito ao domínio de aplicação.

Os autores anteriormente citados procuram explicar, na bibliografia correspondente, a natureza de uma especificação. A exposição, a ser feita aqui, seguindo os referidos autores, será mais rica em detalhes do que seria necessário para os propósitos desta secção. A vantagem de tal exposição alongada está na possibilidade de formar uma idéia completa de todo o desenvolvimento de software. Todavia, tal exposição trará junto a si alguns passos ou procedimentos que não podemos considerar como estritamente necessários do ponto e vista da diversidade a que eu me referi. No final desta secção eu pinçarei os passos que considero necessários no processo.

Existe uma relação importante entre um programa e a sua especificação: o programa deve satisfazer a especificação a partir da qual ele foi construído.

O seguinte esquema dá uma idéia global do processo de desenvolvimento de software:



Para resolver a tarefa que aqui me proponho passo a examinar mais

detalhadamente as duas "pernas" do modelo anterior.

Comecemos com a perna esquerda.

Criar uma especificação formal envolve a criação de uma teoria axiomática, ou axiomatizável, (junto com um aparato lógico). A construção de uma especificação é muito semelhante a construção de uma teoria na ciência, como já foi dito anteriormente. Usualmente uma teoria é gerada para dar conta de algum fenômeno (em sentido amplo) que desejamos explicar. Uma teoria empírica é testada por meio de comparações entre as predições da teoria por um lado e as observações empíricas por outro. Uma especificação, tanto quanto uma teoria cientifica, não pode ser provada correta, sendo possível unicamente ampliar o seu grau de confiabilidade através de teste e análise de predições. Conforme o resultado destes testes a teoria deve ser melhorada ou até mesmo inteiramente reformada, o mesmo ocorrendo com a especificação.

O processo de desenvolvimento de software começa quando alguém verbaliza um problema a ser resolvido dentro de um determinado domínio de aplicação. A tal problema usualmente chamamos de conceito de aplicação. Este problema deve ser entendido como um objeto puro e plenamente identificável. A sua verbalização, ao contrário, é em geral informal, ambígua e incompleta.

O passo que vai da verbalização do conceito de aplicação dentro do domínio de aplicação até a especificação é um no qual paulatinamente vão sendo clarificados os elementos do problema inicial, e onde tal problema inicial vai sendo subdividido em sub-problemas constituintes. A cada sub-problema, do conceito de aplicação, identificado, procuramos dar uma formalização parcial de modo a ressaltar os elementos essenciais do sub-problema.

O que se busca realizar neste processo é a abstração dos elementos

essenciais do problema de tal forma que o resultado desta abstração contenha os princípios mais gerais.

Tal processo se desenrola através de uma série sucessiva de representações informais do problema inicial, o conceito da aplicação, e pela respectiva formalização de tais representações, sendo que no final de tal cadeia, que está longe de ser linear, obtemos uma especificação formal que representa o problema abstratamente.

Neste processo além do ponto de partida, que é a verbalização do conceito de aplicação, usamos também uma, ou algumas, teoria(s) acerca dos objetos pertencentes ao domínio de aplicação, de modo tal que a especificação formal deve refletir os conhecimentos aportados por estas teorias como também deve refletir abstratamente o problema que se quer resolver. Pode se dar o caso que as teorias acerca do domínio sejam construídas junto ao processo de obtenção da especificação formal, caso no qual supostamente são esperados um maior número de "bactrackings" entre as representações informais que constituem os passos do processo desta "perna" esquerda.

A cada sub-problema identificado teremos uma formalização parcial de modo tal que ao fim do processo a especificação formal obtida estará estruturada e composta de diversos sub-problemas.

Para que o processo de geração de uma especificação formal seja aceitável, a verbalização do conceito de aplicação deverá ser um modelo da especificação formal, ou seja a especificação deverá satisfazer a verbalização do conceito de aplicação.

Como ocorre no caso de qualquer teoria acerca de um domínio de objetos, a especificação deverá ser validada contra a verbalização do conceito de aplicação. Tal validação será sempre parcial, e será

realizada através de testes e observações que, se não refutarem a especificação, confirma-la-ão parcialmente. Neste caso, a origem das observações é a verbalização do conceito de aplicação.

Espera-se que as especificações tenham duas propriedades: a primeira e esta é um "must", ela dever ser consistente (ou seja deve ser viável); a segunda e esta é somente desejável, ela pode ser incompleta pois se ela é uma descrição abstrata do problema ocorrerá que muitos detalhes não poderão ser decididos, a não ser durante o processo de geração do programa. A especificação enquanto abstração do que deve realizar um programa deve ser genérica e portanto não pode cuidar de detalhes, assim é possível que ao nível da especificação existam sentenças sobre as quais não podemos decidir se ela ou a sua negação pertencem a especificação. Logo é desejável que a especificação seja incompleta dado o seu caráter abstrato.

Passando agora a segunda "perna" do modelo, vamos verificar qual a relação que existe entre um programa e uma especificação formal.

Se um programa deve satisfazer a especificação a partir da qual ele foi construído devemos perguntar o que significa dizer que um programa satisfaz uma especificação. Bem, segundo Maibaum e Turski satisfazer é o mesmo que ser correto segundo a especificação. A este tipo de correção eles chamam de correção semântica (para distinguir da correção sintática, que depende unicamente das regras gramaticais). Para isto devemos conhecer qual o sentido pretendido do programa e verificar se o programa corresponde a este sentido. Pelo menos esta é a ótica prescrita a quem racionalmente aceita que para a construção de um programa nós partimos de um significado pretendido, ou uso pretendido, expressado pela especificação formal.

Deste modo, esperamos que um programa seja capaz de expressar algum

significado, e que possamos verificar se o significado do programa se adecua ao que prescreve a especificação formal.

A proposta de Turski e Maibaum é nitidamente em favor de considerar o significado de uma sentença (ou um conjunto de sentenças) como o conjunto de sentenças verdadeiras por causa da sentença (ou conjunto de sentenças). Logo o fechamento dedutivo da especificação.

Uma questão se impõe: como relacionar este conceito de significado de uma especificação com o conceito de satisfatibilidade de um programa com respeito a uma especificação?

Para responder esta questão devemos observar que temos a priori um significado pretendido para o programa. A formulação de tal significado é justamente a especificação do programa. Segundo Maibaum e Turski, a construção de um programa, que de algum modo corresponde em significado a uma especificação, se deve ao único fato de querermos expressar o mesmo significado utilizando constructos linguísticos diferentes.

As linguagens de programação são meios amplamente empregados para expressar a solução de problemas, tal que esta solução pode ser executada em um computador. Necessitamos, então, no mínimo, usar duas linguagens, durante o processo da perna direita: uma para descrever o problema (a linguagem de especificação), outra para descrever a solução (a linguagem de programação).

Dois pontos de vista são, então, possíveis com relação as linguagens de programação universais: como meio de expressar uma grande classe de computações; ou como um meio de construir a expressão de uma computação.

O ponto de vista de Maibaum e Turski privilegia o aspecto construtivo, expressado na crença de que o processo de desenvolvimento de software é basicamente uma cadeia de transformações linguisticas.

A verbalização do problema é uma descrição do problema. Já o programa é uma prescrição, de uma solução, para um problema. Por outro lado, programas são objetos formais, enquanto as verbalizações de problemas geralmente não o são. Deste modo faz-se necessário encontrar níveis linguisticos que permitam a passagem da descrição para a prescrição, bem como da informalidade a formalidade. O nivel de união é justamente o da especificação formal.

Em geral, uma especificação é incompleta. Ou seja, há casos onde não conseguimos decidir se uma dada proposição S, ou a sua negação, deve pertencer a teoria, de modo tal que devemos considerar então duas teorias diferentes, uma extendida com S a outra com $-S$. Tal incompletude é até desejável, e a isto chama-se de permissividade da especificação. Tal incompletude é desejável porque a especificação, enquanto teoria, é abstrata, não cuidando de detalhes mas sim de princípios gerais.

Já que uma especificação formal enquanto teoria deve poder ser dada por um conjunto de axiomas, que expressam informação/conhecimento acerca de um dominio de aplicação, e que a especificação é a descrição de um problema, e a prescrição para um programa, seus axiomas devem ter a seguinte forma genérica: $(x)\ (P(x) \rightarrow (Ey)\ Q(x,y))$, onde P e Q são fórmulas e P não contém nenhum outro quantificador existencial. O requisito de construtibilidade de um artefato a partir de uma especificação formal implica que para cada axioma do tipo anteriormente mencionado exista uma função recursiva f, tal que $P(x) \rightarrow (f$ *é definido)* $\&\ Q(x,f(x))$, portanto que possamos "skolemizar" a variável y.

Resumindo, uma especificação tem várias propriedades importantes: primeiro, ela é formal; segundo, não existem significados encobertos em qualquer uma das suas partes; terceiro, ela é uma descrição aceitável do problema; quarto, ela é uma prescrição para o programa que resolverá o

problema. Assim sendo, já que um programa é também formal, o que se deseja é que a relação entre a especificação e o programa seja calculável, em algum sentido. Tal sentido é o da correção do programa com respeito a especificação.

Podemos expressar a relação de correção da seguinte forma: *para qualquer A tal que G |- A, corresponde um B tal que Pr |- B,* onde *G* é a especificação (portanto um conjunto de sentenças) e *Pr* é o programa sobre o qual se quer estabelecer a relação de correção com respeito a *G*. A expressão anterior é uma meta-expressão porque nela são feitas referências a dois sistemas linguísticos diferentes daquele em que a própria expressão é feita. Dada a desejável incompletude da especificação, caso uma propriedade não seja uma consequência de *G* pode ocorrer que ou esta propriedade ou a sua negação sejam, excludentemente, consequência de *Pr*, ou que nenhuma das duas seja consequência de Pr.

Usualmente o significado de um programa é expressado dentro da semântica axiomática por *A -> [a]B* onde *a* é um constructo em uma linguagem de programação, e *A* e *B* são fórmulas do sistema linguistico tal que se algum agente executa *a* quando é o caso que *A* então se o programa termina as circunstâncias obtidas satisfazem *B* (na verdade está é uma fórmula da lógica dinâmica que, por sua vez, tem uma fórmula correspondente dentro da semântica axiomática: *{A} a {B}*, eu prefiro a primeira por ser mais elegante).

Algumas fórmulas do tipo anterior expressam o significado das sentenças e constructos básicos de uma linguagem de programação. Estas fórmulas são os axiomas extralógicos da linguagem. Portanto, todo artefato de software sintaticamente correto tem o seu significado expressado por meio de uma fórmula derivada a partir destes axiomas e do

axioma que relaciona os predicados de entrada e saída.

Dado um artefato podemos obter a sua fórmula semântica correspondente. Entretanto, usualmente, são dados ao programador os predicados *A* e *B*, por meio da especificação formal e seus axiomas. O problema que o programador deve resolver é o de encontrar um *a* que satisfaça a fórmula: *A --> [a] B*. *A* e *B* são chamados genericamente de especificação de entrada e saída.

Voltando à questão da relação de satisfação: *seja R -> [b]Q a fórmula que expressa o significado do programa b então dizemos que: b satisfaz G se e somente se sempre que G |- A então existe uma fórmula B que é tradução da fórmula A tal que R -> [b]Q |- B*. Temos, assim, um método formal para raciocinar sobre programas e as consequências de um programa. Por outro lado, tal definição está em acordo com a definição que os autores fazem da palavra "significado" (o conjunto das consequências). Logo, podemos dizer que um programa que satisfaz uma especificação preserva o significado desta especificação, ainda que as duas possam não ter o mesmo significado, sendo o programa algumas vezes "mais rico em informação".

Há dois pontos que os autores fazem questão de salientar, acerca das especificações e dos programas: primeiro, a obtenção do programa se dá através de sucessivos passos de refinamento que inclui a mudança e a extensão de sistemas linguísticos a partir do sistema linguístico da especificação; segundo, todo programa é, também, uma especificação, pois todo passo de refinamento ao produzir uma representação R_{i+1} a partir de R_i produz uma especificação para o passo seguinte.

Neste ponto eu introduzo uma quebra na sequência de exposição. Creio que este é o momento de fazer algumas observações, que aliás conduzem a uma concepção levemente distinta daquelas apresentadas até agora.

No caso das linguagens de programação imperativa, onde as sentenças ordenam a execução de uma ação, creio que devemos reavaliar, em parte, os papéis da especificação e do programa.

Como o nome bem diz, uma sentença imperativa é uma que ordena uma ação. Portanto, não se coloca a questão de saber se a sentença pode ser verdadeira ou falsa. Somente asserções podem ser verdadeiras ou falsas. Um algoritmo é descrito, usualmente, por um conjunto de sentenças imperativas. As linguagens de programação imperativas descrevem uma função, que é computada segundo os passos do algoritmo escrito.

Segundo penso, não podemos comparar diretamente um texto de programa imperativo com uma especificação, pois uma especificação pode ser dita verdadeira ou falsa porque as suas sentenças podem ser ditas verdadeiras ou falsas, o mesmo não ocorrendo com um programa. E' necessário acrescentar predicados de entrada e saída ao texto do programa imperativo, que é obviamente uma expressão formal, para podermos montar uma fórmula, ou sentença assertiva, esta sim passivel de ser analisada quanto a sua verdade ou falsidade!

Por exemplo, para um texto de programa *p* se acrescentamos fórmulas *P* e *Q* de entrada e saída, respectivamente, obtemos a seguinte fórmula: *P -> [p]Q*, que por sua vez é uma fórmula do mesmo tipo daquelas usadas na semântica axiomática. Desta forma estamos aptos a comparar a especificação com o programa assim contextualizado. Na verdade os passos de refinamento produzem o programa e algo mais. Este conjunto programa e algo mais deve ser uma teoria, o que anteriormente haviamos chamado de *Pr*.

Bem, continuando, me pergunto se esta designação - semântica axiomática - não é, até certo ponto, equivocada, pois o que se faz com as

fórmulas da referida semântica, entre outras coisas, é justamente estabelecer o contexto no qual um programa pode ser avaliado contra a sua especificação. Particularmente, eu preferia dizer que a correção de um programa é estabelecida por uma avaliação interteorética, entre a especificação, que como vimos é uma teoria, e entre uma fórmula de entrada/saida que contém o texto do programa imperativo como functor, ou no caso das fórmulas acima como função de acessibilidade (toda função é relação) entre estados. Melhor seria dizer que semânticamente um programa representa uma função, como é o caso da semântica denotacional.

Tomemos por exemplo uma sentença da linguagem Pascal: *a:=4*5;*. Suponhamos que tal sentença além de sintaticamente correta, pertença a um programa *p* sintaticamente correto. E' fácil verificar que para esta sentença não podemos dizer ser ela verdadeira ou falsa pois ela não é uma asserção. Ela é o que podemos chamar de uma sentença imperativa, a sentença mais básica de uma linguagem de programação para descrever algoritmos.

O meu intuito com as considerações anteriores foi o de marcar a existência de uma diferença essencial entre uma especificação, entendida como uma teoria, e portanto passível de ser expressada em uma linguagem onde as sentenças podem ser ditas verdadeiras ou falsas, e um texto de programa puro e simples, escrito em uma linguagem imperativa.

Uma objeção a este meu argumento é a que aponta para a existêcia de tradutores automáticos que transformam um texto de programa em um outro texto, os compiladores, e que portanto devem manter a correção entre *p*, fonte, e *p'*, objeto, logo devem poder calcular a correção entre *p* e *p'*. Ora, neste caso eu tenho a dizer que os mesmos *P* e *Q* de entrada e saida para um program *p* escrito em uma linguagem *L1* valem para *p'* escrito em uma linguagem *L2*, ou seja, que afinal de contas o passo de compilação não

adiciona elementos relevantes para o que os autores anteriormente haviam chamado de significação. Enfim, no passo de compilação não há propriamente nenhuma tarefa criativa. Portanto os dois programas são equivalentes por tradução mecânica.

Todas estas discussões suscitam uma outra questão: a da interpretação da palavra significado. Para Maibaum e Turski o significado de uma sentença (ou sentenças) é o conjunto das suas consequências. Esta tese, digamos sintaxista, também foi sustentada por Carnap no seu "Logical Syntax of Language". Posteriormente, em um texto junto com Bar-Hillel, Carnap passou a chamar de conteúdo de informação ao conjunto de consequências de uma sentença (ou sentenças). Eu prefiro esta última designação.

Dando continuidade a exposição do assunto, resta verificar em que sentido é usada a palavra calculável quando os autores se referem a relação existente entre uma especificação e um programa que a satisfaz. Para tal eles fazem uma comparação com a prova de um teorema.

Todo e qualquer passo de uma dedução formal é calculável pois todo e qualquer passo segue regras precisas e definidas para a sua confecção. Entretanto a prova mesma do teorema nem sempre pode ser dada por um algoritmo. Empregamos o que se chama de heurística para desenvolver tal prova.

O mesmo acontece com os passos da construção de um artefato de software a partir da sua especificação. Deste modo, como a natureza de cada passo simples é verificável, então toda a sequência de passos é verificável, ou calculável. A estes passos de construção dá-se o nome, como eu já disse, de passos de refinamento.

Os objetos de um passo de refinamento seja ele o objeto de chegada

ou de partida são ambos especificações. Logo cabe perguntar qual é a natureza destes passos. Em primeiro lugar, e isto já sabemos, estes passos adicionam detalhe e, portanto, tornam a especificação menos permissiva. Segundo, esta tarefa de adição de detalhe muitas vezes deve ser acompanhada da mudança ou da extensão do meio linguistico utilizado para expressar a especificação, nas palavras dos autores tradução e extensão teoréticas.

Tal passo de uma especificação a outra é chamado de passo canônico. O passo canônico, ou implementação, é dado por uma dupla $\langle I_i, e_i \rangle$ que leva de uma especificação $i-1$ a uma especificação i, onde $e_i: E_{i-1} < E'_i$ é uma extensão conservativa da especificação $i-1$ na especificação i, e $I_i: E_{i-1} \rightarrow E_i$ é uma interpretação entre teorias (ou tradução entre teorias).

O processo de implementação resume-se basicamente a duas ações: representar os dados de uma especificação em termos da outra; definir operações e testes, correspondentes a aqueles definidos na especificação de partida, em termos de operações e testes da especificação de chegada. Assim a partir da composição de vários passos canônicos temos uma cadeia de implementação.

Quando um programa é refinado os dados com os quais o programa opera também devem ser refinados, ou seja devem ser refinados em paralelo. Estes passos de implementação adicionam um tipo especial de detalhe: a representação da estrutura de dados concomitante com a adição de detalhes algorítmicos correspondentes a esta estrutura de dados.

Assim, finalmente, o programa escrito em uma linguagem formal é uma expressão formal, capaz de ser executado em uma máquina objetivo, que por sua vez fornece a interpretação dos axiomas extralógicos básicos da linguagem.

A máquina virtual daí resultante é um modelo da especificação

formal, e deve satisfazer o conceito de aplicação que deu origem ao processo. Neste caso Haeberer e Veloso dizem que a máquina virtual é um modelo de engenharia do conceito de aplicação.

Como o meu objetivo neste texto é o de descrever o processo de construção de um artefato de software, creio que cabem algumas observações finais, todas de minha autoria.

Tendo em conta a diversidade da atividade de construir software não podemos afirmar propriamente que os passos de refinamento e as representações parciais? anteriores a especificação, são tarefas canônicas segundo o sentido que eu usei no começo do capítulo. Pode ocorrer que nem mesmo a especificação seja formalmente expressada antes da construção do software, ou que todos os passos de refinamento estejam colapsados em um único passo onde o texto do programa é escrito diretamente. Ainda que tais ações possam se desaconselháveis segundo as regras de boa conduta para o desenvolvimento de software, existe quem assim proceda.

Porém, um ponto é essencial: o detalhamento. Todo programa é desenvolvido com a adição de detalhes ainda que não haja nenhuma distinção de nível entre os detalhes adicionados.

Duas outras tarefas, além do detalhamento, são, segundo creio, necessárias: a atividade de teorizar acerca de um objeto e do domínio no qual ele se encontra, pois um software é desenvolvido para resolver um problema, e a compreensão do que é um problema e do que pode ser uma solução dependem do conhecimento organizado de que se dispõe acerca do domínio do problema; e a tarefa de passar da informalidade a formalidade em algum momento do processo de construção de um artefato de software.

A passagem da informalidade a formalidade pode se dar no nível da

teoria, quando, agindo metodicamente, o engenheiro gera uma especificação formal. Quando não agindo tão metodicamente o engenheiro colapsa a geração do programa com a geração da especificação formal, também há uma passagem da informalidade à formalidade, pois o texto do programa é uma expressão formal. Neste último caso pode ocorrer que o programa seja de difícil compreensão. Entretanto dado o texto do programa, que é uma expressão formal, podemos construir a fórmula da semântica axiomática que contextualiza tal programa a partir dos axiomas extralógicos da linguagem, e obter assim uma teoria formalizada. Houve neste caso uma formalização parcial, mas plenamente suficiente para determinar o resto da teoria formalizada. Em particular o engenheiro que construiu tal programa deverá poder fazer esta contextualização, pois no final das contas ele conhece a pré e a pós-condição do programa. Particularmente, no caso de um texto de programa na linguagem Prolog a formalização se dá ao nível da teoria. Um programa Prolog não é um texto imperativo, nele expressamos sentenças sob a forma de cláusulas e regras, logo nele podemos exprimir quase que diretamente a especificação sob forma de teoria.

Retomando a questão que motiva esta dissertação: se qualquer tentativa de produzir um comportamento inteligente em um computador implica, no final das contas, na construção de uma máquina virtual, em particular implica na construção racional de um software, duas tarefas são essenciais: teorização sobre um objeto, no caso sobre o comportamento racional, ou inteligente; passagem da informalidade à formalidade, em outros termos, construção de um programa que é uma expresão formal, mas que sobretudo caracteriza uma regra de um sistema formal.

Se realmente é necessário construir um modelo de engenharia para obter um comportamento inteligente em um computador, o que implica

sobretudo a formalização de uma teoria para a consequente geração de uma máquina virtual, então podemos levantar uma objeção a quem sustenta a possibilidade de construir um modelo de engenharia para todo comportamento racional. Tal objeção será apresentada na conclusão deste trabalho.

D- Referências Bibliográficas

[1] Bird, R.; PROGRAMS AND MACHINES, John Wiley & Sons, 1976

[2] Casanova, M.A. et alli; PROGRAMACAO EM LOGICA E A LINGUAGEM PROLOG, Ed. Edgard Blücher, 1987

[3] Gordon, M.J.C.; THE DENOTATIONAL DESCRIPTION OF PROGRAMMING LANGUAGES, Springer-Verlag, 1979

[4] Haeberer, A.M. e Veloso, P.A.S.; WHY SOFTWARE DEVELOPMENT IS INHERENTLY NON-MONOTONIC: A FORMAL JUSTIFICATION, PUC-RJ, 1989, não publicado

[5] Haeberer, A.M. et alli; FORMALIZACION DEL PROCESSO DE DESARROLLO DE SOFTWARE, Editorial Kapeluz, 1989

[6] Hopcroft, J.E. e Ulmman, J.D.; FORMAL LANGUAGENS AND THEIR RELATION TO AUTOMATA, Addison-Wesley, 1969

[7] Mili, A. et alli; FORMAL MODELS OF STEPWISE REFINEMENT OF PROGRAMS em: ACM Computing Surveys, v. 18, nro. 3, setembro de 1986

[8] Turski, W.M. e Maibaum, T.S.E.; THE SPECIFICATION OF COMPUTER PROGRAMS, Addison-Wesley, 1987

[9] Veloso, P.A.S.; OUTLINES OF A MATHEMATICAL THEORY OF GENERAL PROBLEMS, em: Philosophia Naturalis, v. 21 (2-4), p. 354-367, 1984

CONCLUSÃO

A- Argumentos Contra o Mecanicismo da Razão

Nesta conclusão eu pretendo apresentar um resultado que, acredito, impõe limites ao uso dos sistemas formais, e que parece ter implicações importantes para as pretensões de I.A., e para explicação mecanicista da racionalidade. Para tal exposição eu vou me apoiar nas idéias de J.R. Lucas, expostas no artigo "Minds, Machines and Gödel" ([3]), e no livro "The Freedom of the Will" ([4]) por um lado, e nas idéias de M. Dummett no artigo "The Philosophical Significance of Gödel's Theorems" ([2]) por outro.

A argumentação que eu apresentarei é muito semelhante a aquela argumentação apresentada por J.R. Lucas nos seus escritos. Entretanto, eu não farei nenhuma referência a polêmica gerada a partir de tais escritos, limitando-me somente a apresentar o argumento central que visa mostrar a insustentabilidade da tese da explicação mecanicista-formalista da razão, objetivo desta dissertação.

Passemos a apresentação propriamente dita.

Seja GR a fórmula de Gödel para a aritmética formalizada, modificada por Rosser (GR= [5] pág. 160), e seja *A* a aritmética formalizada dos números naturais:

Teorema C.1- Se *A* é consistente, então GR e -GR são ambas não-demonstráveis em *A*, logo *A* contém uma sentença indecidível ([5] pág. 160).

Teorema C.2- Toda extensão consistente recursivamente axiomatizável

de *A* está sujeita ao teorema *C.1*, e portanto tem uma sentença indecidível ([5] pág. 162).

Podemos concluir a partir dos teoremas anteriores que:

1) Qualquer sistema formal suficiente para expressar a aritmética também é incompleto e recursivamente incompletável;

2) Nós, na qualidadede de seres racionais e inteligentes, podemos reconhecer que a fórmula G é verdadeira.

Devemos observar que a racionalidade como um atributo humano nos é indispensável para aceitar as conclusões anteriores extraídas dos teoremas de Gödel: primeiro, porque nós somos racionais nós podemos reconhecer os teoremas de Gödel como teoremas, dado que podemos compreender e aceitar a construção que nos permite chegar aos referidos teoremas; segundo, e mais fundamental, porque nós somos racionais é que nós podemos reconhecer a fórmula G como verdadeira.

O objetivo de J.R. Lucas, na bibliografia citada anteriormente, é o de demonstrar a insustentabilidade do determinismo físico tomando como ponto de partida os teoremas de Gödel. Meu objetivo aqui é o de apresentar os mesmos argumentos para mostrar a insustentabilidade de uma explicação da razão que eu chamo de mecanicista-formalista, por conseguinte mostrar a insustentabilidade "no atacado" das pretensões de I.A.. "No atacado" porque a objeção diz respeito a impossibilidade de pretender replicar todo e qualquer tipo de comportamento racional que os seres humanos possam ter, o que não significa que muitos comportamentos racionais não possam, efetivamente, ser replicados.

O argumento tem uma forma dialética, pois ele pressupõe a existência de um oponente argumentando a favor da explicação mecanicista-formalista da razão. Logo, os argumentos a serem apresentados são esquemáticos e genéricos de tal forma que sempre que alguém queira sustentar a hipótese

do mecanicismo este alguém seja empurrado para uma posição que eu creio insustentável.

O esquema é o seguinte:

1) O mecanicista, ou determinista, afirma que a razão, ou a inteligência, ou a mente pode ser explicada por meio de um sistema de regras, o que equivale a afirmar que existe um sistema formal suficientemente complexo que descreve o comportamento racional;

2) O contra-argumento é lançado:

a) O mecanicista deve pretender que o sistema seja consistente;

b) Por outro lado o sistema deve ser capaz de dar conta de qualquer coisa que venha a ser entendida como uma capacidade da razão mesmo que não possamos atribuir esta capacidade a todo ser racional;

c) distinguir o verdadeiro do falso é seguramente uma das capacidades da razão;

d) Qualquer sistema formal suficiente para expressar aritmética é incompleto, ou seja é da essência do conceito de sistema formal a característica de não poder determinar como verdadeira uma sentença que nós racionalmente podemos;

e) Logo o modelo mecanicista-formalista da razão apresentado parece ser insustentável, pois não consegue diferenciar o verdadeiro do falso em um caso onde alguns seres racionais conseguem, justamente pelo uso da sua razão;

3) O mecanicista reage e afirma que pode modificar o sistema formal de modo tal que o novo sistema passe a reconhecer como verdadeira a sentença que antes não era reconhecida como verdadeira, ou seja acrescenta a sentença ao conjunto de axiomas;

4) Réplica: ainda assim podemos construir racionalmente outra

sentença de tal forma que o sistema melhorado não pode reconhecer a sua verdade e nós podemos racionalmente fazê-lo;

Conclusão: se aritmética formalizada é consistente então qualquer sistema formal que possa ser suficiente para dar conta do comportamento racional é intrinsecamente incapaz de realizar uma tarefa que nós racionalmente podemos realizar, pois tal sistema deve incluir a aritmética para os números naturais. O jogo dialético pode se extender indefinidamente. Enquanto o mecanicista sustentar a possibilidade de um modelo mecânico sempre há a possibilidade de por o referido modelo em "xeque".

Segundo Dummett isto ocorre justamente porque o significado de ser um número natural é algo inerentemente vago. O conceito de número natural é inerentemente vago porque para qualquer caracterização do conceito uma noção central desta caracterização é a da validade da indução para qualquer propriedade bem definida. Por seu turno o conceito de propriedade bem definida para os números naturais é inerentemente vago pois o conceito de propriedade bem definida é indefinidamente extensível. Um conceito é indefinidamente extensivel se para qualquer caracterização dele há uma extensão natural mais inclusiva. Particularmente, em um sistema formal só podemos incorporar a validade da indução para propriedades representáveis no sistema. Por exemplo, a consistência da aritmética formalizada é uma propriedade bem definida mas não expressável no sistema.

Dummett ao afirmar que o conceito de número natural é inerentemente vago não quer dizer que com isso não possamos compreender o que é um número natural. Ao contrário, podemos perfeitamente saber o que é um número natural, qualquer um deles é obtido a partir do zero por uma quantidade finita de aplicação da operação sucessor. Segundo as palavras

do autor, não há a menor possibilidade de confundir Júlio Cesar com um número natural. O que acontece com a caracterização dada por meio de um sistema formal para os números naturais é que tal caracterização não consegue excluir outros objetos que não são números naturais. Sempre é possível produzir uma caracterização mais estreita do que vem a ser um número natural por adição das fórmulas de Gödel ao conjunto dos axiomas, mas é impossível produzir uma caracterização exata de uma vez por todas.

Ainda segundo Dummett, o problema da incompletude para a aritmética formalizada está na necessidade de pressupor a consistência global do sistema para poder fazer uma inferência de uma série infinita de premissas a uma fórmula com um quantificador universal. O teorema de Gödel tem a seguinte forma: (x) A(x). Mas o predicado "A" e' construído de tal modo que para qualquer "n" número natural nós podemos decidir se A(n) é verdadeiro ou falso. Na verdade para qualquer "n" que seja um número natural A(n) é verdadeiro, pois caso contrário existiria uma demonstração para a fórmula (x) A(x), e portanto o sistema seria inconsistente. O autor afirma que para fazer a demonstração de uma fórmula não necessitamos saber se o sistema é consistente ou não. Tudo o que precisamos são premissas e regras de demonstração intuitivamente aceitáveis. Porém no caso da prova de (x) A(x) a necessidade de prova da consistência é real pois necessitamos poder afirmar um número infinito de premissas (A(1), A(2), ..., A(n), ...). Para o autor o problema em concluir (x) A(x) não está na passagem de um número infinito de premissas para a referida fórmula, mas sim na afirmação da premissa infinita A(1) & A(2) & ....

Quanto a questão da consistência do sistema penso que temos muitas razões para acreditar na mesma. Primeiro porque há uma prova da

consistência que unicamente pode não ser considerada como estritamente finitária, a prova de Gentzen, e que é uma que se baseia na validade da indução para qualquer propriedade bem definida para os números naturais, segundo Dummett. Segundo porque podemos fornecer modelos não-standard para a aritmética formalizada.

Se o que eu expus até agora é realmente correto, podemos ter uma explicação que parece adequadamente mostrar porque um modelo para a razão baseado em um sistema formal é inerentemente menos do que a razão humana. O fato é que ao definirmos um sistema formal a quantidade de propriedades bem definidas representáveis neste sistema formal é em certo sentido limitada, de modo tal que, dada a vaguidade inerente ao conceito de número natural, existem propriedades que não podem ser representadas no sistema formal dado, enquanto para a razão a extensibilidade do conjunto de propriedades bem definidas não causa nenhum problema. Em outras palavras, para extendermos o conjunto de propriedades bem definidas para os números naturais representáveis em um sistema formal devemos modificar o sistema formal, enquanto para a razão tal capacidade de extender o conjunto de propriedades bem definidas parece ser natural: a razão não passa a ser uma outra razão só porque houve tal extensão.

Articulando então o esquema de argumentação anterior com as duas teses defendidas no corpo desta dissertação, respectivamente no capitulo III e no capítulo IV, o argumento completo fica assim:

1) Primeira tese: os computadores são máquinas de sistemas formais. Isto significa que todo o comportamento que um computador possa ter, tal comportamento deve poder ser descrito por um sistema formal. Isto equivale a dizer que o conjunto dos comportamentos do conjunto de todas as máquinas de Turing devem poder ser descritos por um sistema formal. Logo se os sistemas formais são inerentemente deficientes na sua

capacidade de reconhecer como verdadeira uma sentença que nós racionalmente podemos reconhecer como verdadeira, então todo e qualquer comportamento que um computador porventura possa apresentar está sempre aquém do comportamento racional que nós seres humanos podemos apresentar.

2) Segunda tese: toda atividade de programar, ou seja de dar um comportamento para um computador, envolve duas tarefas essenciais: teorização acerca de um determinado domínio e formalização dos elementos desta teorização. Portanto toda atividade de programar, por conseguinte todo modelo mecânico computacional do comportamento racional, construído, justamente por implicar a necessidade de uma formalização, ou seja a construção de um sistema formal, é intrinsecamente limitado: não pode reconhecer como verdadeira umasentença que alguns de nós racionalmente podemos.

Se as conclusões anteriores são verdadeiras então a hipótese do sistema físico de símbolos parece não ser empírica: mais que isso ela é falsa. A hipótese do sistema físico de símbolos falha, segundo creio, porque a noção de máquina de sistemas formais engloba o conceito de sistema físico de símbolos.

B- A Questão da Mente

Esta secção final deve ser considerada mais como um apêndice do que propriamente como parte da conclusão. Nela eu tento mostrar o porque eu uso as palavras "comportamento racional" que certamente o leitor atento terá achado um tanto estranha.

Se as conclusões anteriores são corretas podemos tirar algumas conclusões no que diz respeito a um tópico da filosofia muito trabalhado nos nossos dias: a filosofia da mente

A tese principal da filosofia materialista da mente, e por isso muito debatida, é a que afirma que deve haver uma correspondência entre os estados mentais e os estados cerebrais ([1] pág. 2).

Como foi possivel perceber eu não utilizi dentro desta dissertação a palavra "mente". Eu preferi usar as palavras "comportamento racional" nos contextos em que usualmente se usaria a palavra "mente". Tal preferência deve-se unicamente ao fato da palavra "mente" ser utilizada algumas vezes de um modo com o qual não concordo. Não afirmo que o conceito expressado pelas palavras "comportamento racional" seja melhor definido que o conceito expressado pela palavra "mente", em absoluto. O que ocorre é que, a meu ver, as primeiras palavras são preferiveis a última por serem menos carregadas de significados e usos. Particularmente, eu não acredito na tese dita materialista da filosofia da mente, e portanto não gostaria que outros significados espúrios entrassem na discussão que eu apresentei nesta dissertação.

A correspondência pretendida pela filosofia materialista da mente me parece ser insustentável segundo o que ficou estabelecido na secção

anterior. Vou procurar dar as minhas razões para tal crença.

A palavra "estado" é uma que possui profunda conotação mecânica. Um sistema mecânico tem estados. Se conhecemos o seu estado atual, e se conhecemos as leis que regulam o sistema podemos determinar qual o próximo estado. Ainda que possamos legitimamente falar em estados cerebrais, o que devemos compreender é que esta categoria de conceitos, os conceitos mecânicos, são no final das contas insuficientes para explicar uma característica do comportamento racional, a capacidade de distinguir o verdadeiro do falso, conforme a secção anterior. Deste modo se a palavra mente caracteriza entre outras coisas a razão humana das duas uma: ou a palavra estado não pode ser adequadamente aplicada a palavra mente pois há uma incompatibilidade conceitual, ou a palavra mente não pode caracterizar adequadamente aquilo que entendemos por razão.

No meu entender estamos neste ponto frente a duas alternativas: ou os conceitos mecânicos não devem ser tomados como os únicos conceitos científicos ao nosso dispor, ou devemos renunciar a pretensão de explicar cientificamente a mente enquanto fenômeno. Como eu disse anteriormente o mecanicismo científico é uma visão que pretende que a categoria de conceitos científicos se resume a leis de regularidade, em ultima instância em leis de causa e efeito.

Enfim, eu não uso a palavra "mente" no contexto da minha dissertação, quando tal uso poderia ser muito natural, porque não desejo que significados outros que a palavra carrega possam obnubilar as idéias que eu apresento, e por outro lado porque não aceito a tese da filosofia materialista da mente.

B- Bibliografia

[1] Churchland, P.M.; MATTER AND CONSCIOUSNESS", Bradford Books, 1984

[2] Dummett, M.; THE PHILOSOPHICAL SIGNIFICANCE OF GODEL'S THEOREMS, em: Truth and Another Enigmas, p. 186-201, Duckworth, 1978

[3] Lucas, J.R,; MINDS, MACHINES AND GODEL, em: Philosophy, v.36, p. 112-127, 1961

[4] Lucas, J.R.; THE FREEDOM OF THE WILL, Claredon Press, 1970

[5] Mendelson, J.R.; INTRODUCTION TO MATHEMATICAL LOGIC, D. Van Nostrand, 1979