Gödel e a Computabilidade [1]

Luís Moniz Pereira

Centro de Inteligência Artificial – CENTRIA
Universidade Nova de Lisboa

As perguntas de Hilbert

O programa de investigação de David Hilbert apresentado no congresso internacional de matemática de Bolonha em 1928 constituía essencialmente uma extensão do trabalho que iniciara na década de 1890. Não tentava responder à questão que Frege e Russell haviam colocado, quanto ao que a matemática *era verdadeiramente*. Nesse aspecto mostrava-se menos filosófico, menos ambicioso. Por outro lado era muito mais penetrante pois levantava questões profundas e difíceis *acerca* dos sistemas como os que Russell produzira. De facto Hilbert fez a pergunta sobre quais eram, por princípio, as limitações de um esquema como o de *Principia Mathematica*. Haveria algum modo de descobrir o que se podia, e o que não se podia, demonstrar adentro de tal teoria?

A abordagem de Hilbert foi chamada de *formalista*, porque tratava a matemática como um jogo, uma questão de forma. Os passos permitidos numa demonstração deveriam ser encarados do mesmo modo que as jogadas possíveis do jogo de xadrez, sendo os axiomas a posição de partida do jogo. Nesta analogia, 'jogar xadrez' correspondia a 'executar matemática', mas as *afirmações* acerca do xadrez (tais como "dois cavalos não conseguem fazer xeque-mate") corresponderiam a afirmações *acerca* do âmbito da matemática. Era com tais afirmações que o programa de Hilbert se preocupava.

Nesse congresso de 1928, Hilbert tornou as suas questões muito precisas. Primeiro, seria a matemática *completa*? No sentido técnico de que toda a afirmação (tal como "todo o inteiro é a soma de quatro números quadrados") podia ser demonstrada, ou rebatida. Segundo, era a matemática *consistente*? No sentido de que a afirmação '2+2=5' nunca poderia ser alcançada por uma sequência de passos válidos de uma demonstração. E terceiro, era a matemática *decidível*? Queria com isto dizer, se existiria um método bem definido o qual, em princípio, ao ser aplicado a qualquer asserção, conseguiria garantidamente produzir uma decisão correcta quanto à verdade da asserção. Em 1928, nenhuma destas questões tinha resposta. Opinava Hilbert que a resposta era 'sim' em todos os casos. Na opinião dele, "não havia problemas irresolúveis." Mas em breve o jovem matemático da Áustria, Kurt Gödel, anunciou resultados que assentavam um sério golpe no programa de Hilbert.

As respostas de Gödel

Kurt Gödel (doravante G) conseguiu mostrar que toda a formalização da aritmética tem de ser *incompleta*: que há asserções que não podem ser provadas nem refutadas. Partindo dos axiomas de Peano para os inteiros, ampliou-os com uma teoria de tipos simples, de modo a que o sistema obtido conseguisse representar conjuntos de inteiros,

[1] Artigo convidado, para um número totalmente dedicado a Kurt Gödel, do Boletim da Sociedade Portuguesa de Matemática (BSPM), a publicar em 2006 por ocasião do centésimo aniversário do seu nascimento.

conjuntos de conjuntos de inteiros, e por aí adiante. Contudo, a sua argumentação aplicava-se a qualquer sistema formal suficientemente rico para incluir a teoria dos números, e os detalhes dos axiomas não eram cruciais.

A seguir mostrou que todas as operações de uma 'prova', as tais regras de dedução 'à xadrez', podiam ser representadas aritmeticamente. Isto é, elas só empregariam operações tais como a contagem e a comparação, para verificar se uma expressão (também ela representada aritmeticamente) havia sido correctamente substituída por outra – tal como o teste de legalidade de uma jogada de xadrez é apenas uma questão de contar e comparar. Na verdade, G mostrou que as fórmulas do seu sistema podiam ser codificadas como inteiros, representando asserções *acerca* de inteiros. Esta é a sua ideia chave. Continuando, G mostrou como codificar demonstrações como inteiros, de forma que tinha toda uma teoria da aritmética, codificada *dentro* da aritmética. Tratava-se de um aproveitamento do facto de que se a matemática era para ser encarada como um puro jogo de símbolos, então já agora empregar-se-iam símbolos numéricos em vez de quaisquer outros. Ele conseguiu mostrar que a propriedade de 'ser uma prova' ou de ser 'demonstrável' era tão aritmética como a de 'ser quadrado' ou de 'ser primo'.

O efeito deste processo de codificação foi o de se tornar possível escrever asserções aritméticas que se referiam *a si próprias*, tal quando uma pessoa diz "Estou a mentir". De facto G construiu uma asserção específica que tinha exactamente essa propriedade, pois que dizia efectivamente "Esta asserção não é demonstrável." Seguia-se que ela não podia ser demonstrada *verdadeira*, pois tal levaria a uma contradição. Nem podia ser demonstrada *falsa*, pela mesma razão. Era pois uma asserção que não podia ser nem provada nem refutada por dedução lógica a partir dos axiomas. Portanto G provara que a aritmética era *incompleta*, no sentido técnico de Hilbert.

Mas há mais que se diga sobre assunto, porque algo de notável acerca dessa asserção especial de G é que, uma vez que não é demonstrável, é em certo sentido *verdadeira*. Mas o dizer que é *verdadeira* requer um observador que seja capaz, digamos, de olhar para o sistema a partir de fora. Não pode ser mostrado *por de dentro* do sistema axiomático.

Um outro ponto é o de que esta argumentação assume que a aritmética é *consistente*. De facto, se a aritmética fosse inconsistente, então *qualquer* asserção seria demonstrável, uma vez que em lógica elementar de uma contradição tudo se segue. Portanto, mais precisamente, G mostrara que a aritmética uma vez formalizada tem que ser ou *inconsistente,* ou *incompleta*[2]. Também conseguiu demonstrar que a aritmética não podia ser provada consistente adentro do seu próprio sistema axiomático. Para provar a consistência, bastaria provar que havia uma proposição (digamos, '2+2=5') que não podia ser demonstrada verdadeira.

Mas G pôde demonstrar que uma tal afirmação de existência tinha a mesma característica que a asserção que afirmava a sua própria não provabilidade. Deste modo, ele «despachou» as duas primeiras questões postas por Hilbert. A aritmética *não* podia ser provada consistente, e de certeza que *não* era consistente *e* completa. Isto era uma

[2] Na verdade, o facto da asserção de Gödel não poder ser refutada formalmente depende da omega-consistência do sistema, não somente da sua consistência. Mais tarde Rosser, com uma asserção um pouco diferente, mostrou que é possível prescindir da omega-consistência.

reviravolta espantosa na investigação sobre o assunto, e perturbador para os que pretendiam que a matemática fosse absolutamente perfeita e inatacável.

Gödel, computabilidade, e Turing

A *terceira* das questões de Hilbert, a da *decidibilidade*, permanecia em aberto, embora agora tivesse que ser equacionada em termos de *provabilidade* em vez de *verdade*. Os resultados de G não eliminavam a hipótese de que houvesse alguma maneira de distinguir as afirmações demonstráveis das não demonstráveis. Porventura as peculiares asserções auto-referentes de G poderiam ser de algum modo separadas das restantes. Existiria um *método* bem definido, i.e. um procedimento *mecanizável*, que pudesse ser aplicado a uma qualquer afirmação matemática, e dissesse se ela era ou não derivável num sistema formal?

De 1929 a 1930, G resolvera já a maioria dos problemas fundamentais levantados pela escola de Hilbert. Uma das questões que permanecia era a de encontrar um conceito preciso que caracterizasse a noção intuitiva de computabilidade. Mas não era claro na altura que o problema admitisse uma resposta definitiva. Provavelmente G foi surpreendido pela solução dada por Turing, a qual era mais elegante e conclusiva do que esperava. G percebe bem, no início dos anos 30, que o conceito de sistema formal está intimamente relacionado com o de procedimento mecanizável, e considera o trabalho de Alan Turing de 1936 (sobre números computáveis) como um importante completamento do seu próprio trabalho sobre os limites da formalização.

Em 1934, G fez palestras no Institute of Advanced Studies de Princeton onde recomendava a análise de Turing sobre procedimentos mecanizáveis (publicada em 1936) como um avanço essencial que elevava os seus teoremas de incompletude a uma forma mais acabada. Em consequência do trabalho de Turing esses teoremas podem ser vistos a "aplicarem-se a *qualquer* sistema formal consistente contendo parte da teoria finitária dos números".

É conhecido que G e Alonzo Church discutiram, provavelmente no início de 1934, o problema de se encontrar uma definição precisa para o conceito intuitivo de computabilidade, um assunto sem dúvida de grande interesse para G. De acordo com Church, G levantou especificamente o tema da relação entre esse conceito e a sua definição de *recursividade*, mas não achava que as duas ideias se pudessem satisfatoriamente igualar, "excepto heuristicamente". Ao longo dos anos G creditou regularmente o artigo de Turing de 1936 como a obra definitiva que captura o conceito intuitivo de computabilidade, por ser este o único autor a avançar argumentos persuasivos quanto à adequação do conceito preciso assim definido.

Relativamente ao conceito de procedimento mecanizável, os teoremas de incompletude de G clamavam também naturalmente por uma definição exacta (tal como Turing viria a conseguir) pela qual se pudesse dizer que se aplicavam a todo o sistema formal, i.e. todo o sistema acerca do qual as provas se podem verificar com um procedimento automático. Na verdade, o programa de Hilbert incluía o *Entscheidungsproblem* (literalmente, 'problema da decisão'), que pretendia um procedimento para decidir se, em lógica elementar, uma qualquer proposição era derivável ou não pelas regras de Frege (para a lógica elementar, também dita de primeira ordem). Tal requer um conceito preciso de procedimento automático, caso a resposta seja negativa (como é o caso).

Para este efeito, em 1934, G introduz o conceito de funções recursivas gerais, que, depois, se veio a mostrar depois capturarem o conceito intuitivo de computabilidade mecanizável. G sugeriu o conceito e Kleene trabalhou-o. A génese do conceito de funções recursivas gerais estava implícita na prova de G da incompletude da aritmética. Porque quando G mostrou que o conceito de prova através de regras "à xadrez" era um conceito 'aritmético', ele estava na verdade a dizer que uma prova podia ser realizada por um método 'bem definido'. Esta ideia, uma vez formalizada e um tanto estendida, levou à definição de 'função recursiva'. Veio depois a verificar-se que estas eram exactamente equivalentes às funções computáveis.

Mais tarde, G veio a reconhecer que a concepção de 'máquina de Turing' fornece a definição mais satisfatória de 'procedimento mecânico'. Efectivamente, de acordo com o próprio G, é o trabalho de Turing de 1936 sobre números computáveis que apresenta pela primeira vez uma análise convincente dum tal procedimento, mostrando a perspectiva correcta pela qual se pode perceber de modo claro o conceito intuitivo.

Acerca da definição rigorosa do conceito de computabilidade feita por Turing, diz G: "Esta definição de certeza absoluta que não foi supérflua. Contudo, se à partida na definição o termo 'mecanicamente calculável' não tivesse um significado claro, embora não analisado [ou sintético], a questão quanto ao adequamento da definição de Turing não teria sentido, mas, ao invés disso, com Turing recebe indubitavelmente uma resposta afirmativa".

Uma vez que o conceito rigoroso, tal como definido por Turing, seja aceite como o correcto, basta então um pequeno passo para ver que, não só que os teoremas de incompletude de G se aplicam aos sistemas formais em geral, como também para mostrar que o *Entscheidungsproblem* é irresolúvel. A demonstração desta irresolubilidade pelo próprio Turing mostrou que uma classe de problemas não solucionável pelas suas "A-machines" (de "**A**utomatic machines", hoje conhecidas por máquinas de Turing) pode ser expressa por proposições em lógica elementar.

Na sua tese de Ph.D., Turing interessa-se por procurar um escape para a força do teorema da incompletude de G. A ideia fundamental era a de adicionar ao sistema de partida sucessivos axiomas. Cada asserção 'verdadeira mas não demonstrável' é adicionada como novo axioma. Mas, deste modo, a aritmética adquire a natureza de uma Hidra, pois que, uma vez adicionado o novo axioma, nascerá nova asserção daquele tipo que já leva este em conta. Não basta assim adicionar um número *finito* de axiomas, é preciso adicionar infinitos, o que ia contra o sonho finitário de Hilbert. Se se conseguisse ter um gerador finito de tais axiomas, então a teoria inicial seria finita e portanto sujeita ao teorema de G.

Uma outra questão é a de que existe um número infinito de ordens possíveis para acrescentar tais axiomas, conduzindo a teorias diferentes mais completas. Turing descreveu as suas diferentes extensões aos axiomas da aritmética como 'lógicas ordinais'. Nestas, a regra para gerar os novos axiomas é dada por um 'processo mecânico' que se aplica às 'fórmulas ordinais', mas o determinar se uma fórmula é 'ordinal' *não* era mecanizável. Turing comparou a identificação de uma fórmula 'ordinal' ao trabalho da intuição, e considerou os seus resultados como desanimadoramente negativos, pois embora existissem 'lógicas completas' elas sofriam

do defeito de não se poder contar quantos os passos 'intuitivos' necessários para provar um teorema[3].

No entanto, este seu trabalho teve um efeito lateral agradavelmente duradouro, que foi a introdução do conceito de 'máquina de Turing com oráculo', precisamente para poder perguntar e obter do exterior a resposta a um problema irresolúvel adentro dela (como o da identificação de 'fórmula ordinal'). Tal introduziu a noção de computabilidade relativa, ou de irresolubilidade relativa, que abriu um novo domínio na lógica matemática, e na ciência da computação.

O relacionamento, feito por S.A. Cook em 1971, entre máquinas de Turing e o cálculo proposicional viria a dar origem ao estudo de questões centrais de complexidade de computação. Em 1936, também G fazia notar que teoremas com provas longas num dado sistema podem obter provas bem mais curtas em extensões ao sistema. Esta ideia tem sido aplicada em Informática para demonstrar a ineficiência de alguns procedimentos de decisão conhecidos, no sentido de provar que requerem um tempo exponencial (ou pior) para serem executados. Este é um caso particular duma ideia genérica favorita de G: problemas surgidos num dado sistema podem ser tratados com mais resultados num sistema mais rico.

Mens ex-machina

Cabe mencionar que, ao contrário de Turing, G não se interessou pelo desenvolvimento dos computadores. Ora o funcionamento destes está tão relacionada com as operações e conceitos da lógica que, hoje em dia, é vulgar os lógicos envolverem-se, de uma maneira ou de outra, no estudo dos computadores e da informática. No entanto, o famoso teorema de G, dito da incompletude, demonstra e estabelece a inexauribilidade da matemática e a limitação dos sistemas formais e, segundo alguns, a dos programas de

---

[3] Aqui gostaria de fazer referência a trabalho meu em curso. A ideia de base é a de usar uma linguagem de programação em lógica implementada, o EVOLP (Alferes et al. 2002), que tem capacidade de auto *updating*, para obter a tal auto evolução preconizada por Turing.

A teoria de modelos pode definir-se com a semântica revista de modelos estáveis (Pereira e Pinto, 2005), com a introdução em cada programa do seguinte esquema clausal, envolvendo o usual constructo *not* da programação em lógica, que representa a não provabilidade:

G ← asserção_gödel(G), *not* G

Este esquema, representando as suas instâncias *ground*, afirma que se G é uma asserção de Gödel, e não é demonstrável, então G é verdadeira, onde se pressupõe dada a definição do predicado asserção_gödel(G) que testa, uma vez codificada, se o literal G é construível segundo o método de Gödel.
A conclusão é obtida por uso de redução ao absurdo, i.e. se *not* G fosse verdade então G seria verdade; como tal é uma contradição e a lógica é a dois valores, a premissa *not* G é falsa e a conclusão G é verdade.

O sistema de prova parte desta intuição, e utiliza a seguinte regra EVOLP, onde o literal G pode ser por sua vez uma regra:

assert(G) ← asserção_gödel(G), *not* G

Ou seja, cada G obedecendo às premissas fará parte do estádio seguinte do programa auto evolutivo, e assim sucessivamente. Tal regra toma o lugar do passo intuitivo necessário à evolução, realizando-o conceptualmente por uma redução ao absurdo seguido de um *update*, o qual é permitido ser expresso dentro da própria linguagem EVOLP.

computador. Relaciona-se pois com a conhecida questão sobre se a mente ultrapassa a máquina. Assim, a cada vez maior atenção prestada aos computadores e à Inteligência Artificial (IA) tem levado a um aumento geral do interesse pelo trabalho de G. Mas, como G ele próprio reconhece, o seu teorema não resolve a questão de se a mente se sobrepõe à máquina. Aliás, o trabalho de G nesta direcção parece evidenciar uma posição a favor (ao invés de contra) do *mecanicismo* (e até do *finitismo*) como abordagem à automatização dos sistemas formais.

G contrasta *intuição* com *prova.* Uma prova pode ser explícita e conclusiva porque tem o suporte de axiomas e regras de inferência. Em contraste, as intuições podem ser comunicadas apenas através do apontar para as coisas. Toda a filosofia que se expressa como uma teoria exacta pode ver-se como um caso especial da aplicação do realismo conceptual de G. Segundo o próprio, ela deve pretender trazer-nos a perspectiva correcta de modo a vermos claramente os conceitos metafísicos de base. Mais explicitamente, G diz que essa tarefa consiste em determinar com a intuição os conceitos primitivos *C* da metafísica e em encontrar para eles os axiomas *A* (tal que apenas os *C* satisfazem *A*, e tal que os *A* são implicados pela nossa intuição original de *C*). Admite ainda que, de tempos a tempos, possamos adicionar novos axiomas.

G advoga também um 'racionalismo optimista'. A sua justificação para isso apela a (1) "O facto de aquelas partes da matemática que têm sido sistemática e completamente desenvolvidas … mostrarem um espantoso grau de beleza e perfeição." Portanto (2) Não se dá o caso "de que a razão humana é irremediavelmente irracional ao colocar-se perguntas a que não consegue responder, ao mesmo tempo que assevera enfaticamente que só a razão lhes pode responder." Segue-se que (3) Não existem "questões da teoria dos números indecidíveis para a mente humana." Portanto (4) "A mente humana ultrapassa todas as máquinas."

Mas a inferência de (1) para (2) parece obter-se a partir de sucessos acidentais em áreas muito limitadas para justificar uma antecipação de sucesso universal. Além disso, ambas (2) e (3) dizem respeito só a uma parte delimitada da mente e da razão referentes apenas a questões matemáticas.

G compreende que o seu teorema da incompletude por si só não implica que a mente humana ultrapasse todas as máquinas. É necessária uma premissa adicional. G faz três sugestões com esse fito: (a) É suficiente aceitar o seu 'realismo optimista' (b) Ao apelar "ao facto de que *a mente, no seu uso, não é estática, mas se encontra em constante desenvolvimento,*" G sugere que "não há razão para que" o número de estados mentais "não possa convergir para o infinito no decurso do seu desenvolvimento." (c) Ele acredita que existe uma mente separada da matéria e que tal será provado "cientificamente (talvez pelo facto de não haver células nervosas suficientes para realizar as operações mentais observáveis)."

Há uma conhecida ambiguidade entre a noção de *mecanismo* confinado ao mecanizável (no sentido preciso de computável ou recursivo) e a noção de *mecanismo* materialista. G enuncia duas proposições: (i) O cérebro funciona basicamente como um computador digital. (ii) As leis da física, nas suas consequências observáveis, têm um limite finito de precisão. Ele pensa que (i) é muito provável, e que (ii) é praticamente certa. Talvez a interpretação que G faz de (ii) seja a que a torna compatível com a presença de leis físicas não mecânicas, ao mesmo tempo que a liga com (i) no sentido de que, tanto

quanto conseguimos observar sobre o comportamento do cérebro, ele funciona como um computador digital.

A intuição matemática é algorítmica?

Roger Penrose (1994) diz que *não*, e baseia muito do seu argumento, como antes dele J. R. Lucas (revisitado em 1996), no teorema da incompletude de G: é a *intuição* que nos permite *ver* que uma asserção de G, indecidível num dado sistema formal, é por isso verdadeira. Como é que esta intuição poderia ser o resultado de um algoritmo? Penrose insiste em que o seu argumento teria sido "certamente considerado pelo próprio G na década de 1930 e nunca devidamente refutado desde então…"

Mas na sua palestra Gibbs na *American Mathematical Society* em 1951, G contradiz abertamente Penrose:

"Por outro lado, na base do que foi provado até agora, permanece possível que possa existir (e a ser até empiricamente descoberta) uma máquina demonstradora de teoremas que de facto seja equivalente à intuição matemática, embora não se possa *provar* tal, nem mesmo provar que obtenha apenas teoremas correctos da teoria finitária dos números."

Realmente, durante os anos 1930, G foi ultra cuidadoso a evitar afirmações controversas, remetendo-se ao que podia ser provado. Contudo, a sua palestra de Gibbs foi uma verdadeira bomba. G argumentou com insistência que o seu teorema tinha importantes implicações filosóficas. Mas, a citação acima torna-o claro, nunca afirmou que a intuição matemática se podia mostrar ser não-algorítmica.

É provável que G concordasse com o juízo de Penrose de que a intuição matemática não pode ser o produto de um algoritmo. De facto, G aparentemente acreditava que a mente humana nem sequer poderia ter sido um produto da evolução natural. No entanto, G não afirmava que tais conclusões fossem consequência do seu famoso teorema.

Devido ao actual prevalente clima de restringir a atenção acerca dos limites da racionalidade, por oposição à intuição, aos bem definidos e surpreendentes avanços na tecnologia e programação informáticas, a perspectiva de encarar a razão em termos das capacidades maquinais tem recebido muita atenção nas últimas décadas. Tal é reconhecido como nuclear ao estudo da IA, o que é claramente relevante para o desejo de G de separar mentes e máquinas.

Deste ponto de vista, a IA estaria principalmente interessada no que é factível do ponto de vista da computabilidade, cuja faceta formal envolve apenas uma muito pequena parte da matemática e da lógica. Mas o estudo das limitações da IA não pode ser reduzido a esta restrição do seu assunto. Nesse aspecto é crucial distinguir entre algoritmos *para* resolver um problema, e algoritmos *simpliciter*, como conjunto de regras a seguir de modo sistemático e automático, eventualmente auto-modificáveis, *sem* necessariamente um problema específico bem definido para resolver.

Consciência lógica

Se perguntarmos "Como introduzir o inconsciente nos computadores?" alguns colegas dirão que os computadores são totalmente inconscientes. Na verdade o que não sabemos ainda é como introduzir consciência na algoritmia, porque usamos os computadores como apêndices inconscientes da nossa consciência. A questão é pois prematura pois só nos poderemos referir à acepção humana de inconsciente depois de introduzida a consciência na máquina. Na verdade, compreendemos muito melhor o inconsciente computacional do que o nosso próprio inconsciente.

Estas questões prenhes apontam para a complexidade dos nossos processos de pensamento, incluindo os da criatividade e da intuição, que em boa verdade desconhecemos, e lançam um repto muito mais rico à IA, a qual poderá servir-nos enquanto espelho simbiótico indispensável.

A tradução, numa construção computacional, de um modelo funcional, como aventado acima, de uma consciência introspectiva e portanto auto-referente, seria possível utilizando quaisquer metodologias e linguagens de programação actualmente à nossa disposição. Face a esta perspectiva, poderemos perguntar-nos o porquê de utilizar o paradigma da lógica, via a programação em lógica, que é justamente aquele que preferimos (e.g. Lopes and Pereira (2006), e trabalho em curso). Existem vários argumentos que podem ser levantados contra o seu uso, pelo que tentaremos nesta secção reproduzir os mais relevantes, rebatê-los e apresentar a nossa própria argumentação a favor.

O primeiro argumento geralmente levantado nestas discussões é o de que o raciocínio usual do ser humano não usa lógica, existindo processos não-simbólicos complexos no cérebro que supostamente não podem ser emulados por qualquer processamento simbólico. Seguindo esta linha de pensamento, foram produzidos diversos modelos para a inteligência baseados em redes neuronais artificiais, com propriedades emergentes de sistemas puramente reactivos, e muitos outros, numa tentativa de escapar da tirania da GOFAI ('Good Old Fashioned AI'). Existe no entanto um delicado calcanhar de Aquiles oculto nestes modelos. A verdade é que a sua implementação, pelos respectivos proponentes, acaba por ser feita recorrendo a computadores, que vão utilizar processamento simbólico para simular todos esses paradigmas.

A relação deste argumento com a lógica é assegurada pela perspectiva filosófica funcionalista: a própria lógica pode ser implementada em cima de um sistema de processamento de símbolos e, além disso, o substrato particular de suporte físico não é crucial. Uma vez descrito um processo em lógica, podemos usá-la para sintetizar um artefacto com essas mesmas propriedades. Quer isto dizer que, enquanto estivermos a lidar com modelos computacionais, qualquer tentativa para escapar à lógica não conseguirá provar-se inerentemente mais poderosa.

Por outro lado, existe uma evidente capacidade humana para compreender o raciocínio lógico, uma capacidade desenvolvida no decurso da evolução do cérebro. Actualmente, a sua materialização mais poderosa é a própria ciência, e o conhecimento obtido de numerosas disciplinas, cada uma dotada da sua própria nuance lógica dedicada ao raciocínio dentro do seu domínio. Desde leis políticas até à física quântica, a lógica, em

sentido lato, tornou-se o pilar em cima do qual todo o conhecimento humano é construído e melhorado, a última recompensa do nosso domínio sobre a linguagem.

O homem consegue usar a linguagem sem aprender gramática. No entanto, se queremos entender linguística é crucial conhecer a lógica da gramática, sintaxe e semântica. Os seres humanos usam de facto gramática sem o seu conhecimento explícito, mas isso não quer dizer que ela não possa ser descrita. Da mesma forma, quando falamos sobre o movimento de electrões não dizemos que o electrão em si sabe as leis que regem esse movimento, mas estamos sem dúvida a utilizar uma linguagem simbólica para descrever o processo, sendo inclusive possível utilizar essa descrição para implementar um modelo e uma simulação que exibam precisamente o mesmo comportamento. Do mesmo modo, ainda que a consciência humana não opere directamente sobre lógica, isso não significa que não sejamos forçados a utilizar a lógica para construir uma *descrição* suficientemente rigorosa do processo. E, se for usada a programação em lógica, essa descrição pode funcionar como especificação executável.

Uma vez obtida uma descrição suficientemente rigorosa do sistema da consciência, estamos supostamente na posse de todo o *nosso* (provisório) conhecimento actual, reduzido a uma conexão de caixas negras minimais dentro das quais não sabemos encontrar ainda outros mecanismos essenciais. Até à data ninguém se mostrou capaz de dividir muito a actual caixa negra da nossa consciência sobre a consciência no cérebro, e talvez seja possível apresentar uma descrição suficientemente rigorosa dela de forma a podermos modelar um sistema funcional que lhe seja equivalente. Se mais tarde for conseguida a divisão dessa caixa negra epistémica cerebral em outras tantas, será certamente possível, nesta visão, descrever novos modelos computacionais equivalentes aos seus modelos funcionais.

Na sua luta por essa descrição rigorosa, o campo da IA tornou viável a proposta de transformar a lógica numa linguagem de programação (Pereira 2002). A lógica pode actualmente ser utilizada como uma linguagem de especificação que não só é executável, mas sobre a qual podemos demonstrar propriedades e provas de correcção que validam a própria descrição que produzimos com ela. Ao responder ao desafio, o campo da IA desenvolveu a lógica para além das restrições da cumulatividade monotónica, respondendo às necessidades típicas, longe dos paraísos artificiais do bem definido, mas bem dentro do mundo real do conhecimento incompleto, contraditório, passível de revisão, distribuído e actualizável, que obrigam, entre outros, ao estudo e desenvolvimento de lógicas não-monotónicas, e sua implementação em computador.

Ao longo dos últimos anos tem sido desenvolvido imenso trabalho em tópicos individuais como semânticas de linguagens de programação em lógica, revisão de crenças, preferências, programas evolutivos e em muitos outros problemas que são cruciais para uma arquitectura computacional da mente. Estamos em presença de um estado-da-arte em que podemos começar a endereçar estas problemáticas mais gerais utilizando as ferramentas que se encontram à nossa disposição, unificando tais esforços em poderosas implementações exibindo novas e promissoras propriedades computacionais. A lógica computacional mostrou-se capaz de evoluir ao encontro das exigências das descrições difíceis que procura atingir.

A utilização do paradigma da lógica também viabilizará apresentar a discussão de uma tal arquitectura a níveis de abstracção e generalidade suficientemente altos para permitir

produtivas discussões interdisciplinares tanto acerca da sua especificação como das suas propriedades derivadas.

Como previamente mencionado, a linguagem da lógica é universalmente usada quer pelas ciências naturais, quer pelas humanidades, e mais geralmente no seio de qualquer fonte de conhecimento humano, pelo que nos oferece uma plataforma comum sobre a qual podemos raciocinar acerca das nossas teorias. Uma vez que o campo das ciências cognitivas é essencialmente um esforço conjunto da parte de muitas áreas distintas do conhecimento, acreditamos que tais esforços de unificação de linguagem e vocabulário são não só úteis, mas obrigatórios.

Bibliografia Consultada

J. J. Alferes, A. Brogi, J. A. Leite, L. M. Pereira (2002). Evolving Logic Programs. in: S. Flesca et al. (eds.), Procs. of the 8th European Conf. on Logics in Artificial Intelligence (JELIA'02), 50-61, Springer, LNAI 2424, 2002.

J. L. Casti, W. DePauli (2000). *Gödel – A life of Logic*. Basic Books 2000.

M. Davis (1990). Is mathematical insight algorithmic? *Behavioral and Brain Sciences*, 13 (4), 659- 60, 1990.

M. Davis (1993). How subtle is Gödel's theorem. More on Roger Penrose. *Behavioral and Brain Sciences,* 16, 611-612, 1993.

M. Davis (2000). *The Universal Computer: The Road from Leibniz to Turing*. W.W. Norton & Co. 2000.

D. C. Dennett (2005). *Sweet Dreams – philosophical obstacles to a science of consciousness.* The MIT Press 2005.

C. Glymour (1992). *Thinking things through*. The MIT Press 1992.

A. Hodges (1983). *Alan Turing – the enigma*. Simon and Schuster 1983.

G. LaForte, P. J. Hayes, K M. Ford (1998). Why Gödel's theorem cannot refute computationalism. *Artificial Intelligence* 104, 265-286, 1998.

G. Lopes, L. M. Pereira (2006). Prospective Programming with ACORDA. in: Empirically Successful Computerized Reasoning (ESCoR'06) workshop, at The 3rd International Joint Conference on Automated Reasoning (IJCAR'06), Seattle, USA, 2006.

J. R. Lucas (1996). Minds, Machines, and Gödel: A Retrospect. in: P. Millican and A. Clark (eds.), *Machines and Thought (vol. 1),* 103-124, Oxford University Press 1996.

D. McDermott (2001). *Mind and Mechanism*. The MIT Press 2001.

M. L. Minsky (1967). *Computation: Finite and Infinite Machines*. Prentice-Hall 1967.

E. Nagel, J. R. Newman (2001). *Gödel's Proof*. New York University Press 2001.

E. Nagel, J. R. Newman, K. Gödel, J.-Y. Girard (1989). *Le Théorème de Gödel*. Éditions du Seuil 1989.

R. Penrose (1994). *Shadows of the Mind: a search for the missing science of consciousness.* Oxford University Press 1994.

L. M. Pereira (2002). Philosophical Incidence of Logical Programming. in: *Handbook of the Logic of Argument and Inference,* D. Gabbay et al. (eds.), 425-448, Studies in Logic and Practical Reasoning series, vol.1, Elsevier Science 2002.

L. M. Pereira, A. M. Pinto (2005). Revised Stable Models – a semantics for logic programs. in: C. Bento et al. (eds.), Progress in AI, Procs. 12th Portuguese Intl. Conf. on Artificial Intelligence (EPIA'05), 29-42, Springer, LNAI 3808, 2005.

A. Turing, J.-Y. Girard (1995). *La Machine de Turing*. Éditions du Seuil 1989.

H. Wang (1973). *From Mathematics to Philosophy*. Routledge 1973.

H. Wang (1987). *Reflections on Kurt Gödel*. The MIT Press 1987.

J. C. Webb (1980). *Mechanism, Mentalism and Meta-mathematics*. Reidel 1980.