MICROCOMPUTADOR CURSO PRÁTICO

TIRE O MÁXIMO DO MICRO EM 30 SEMANAS

PUBLICAÇÃO SEMANAL ILUSTRADA

Cr$ 12.900

27

Banco de dados instantâneo • Conversa de robô
RAT, o joypad de controle remoto

rioGráfica

MANAUS, ALTAMIRA, MACAPÁ, JI-PARANÁ, PORTO VELHO, RIO BRANCO, SANTARÉM, BOA VISTA, SINOP, ALTA FLORESTA: Cr$ 16.770

MICROCOMPUTADOR CURSO PRÁTICO

TESTE

Depois de ler com atenção todo o fascículo, responda a estas perguntas. Veja a solução no próximo número, junto com um novo teste.

1) Como é possível a um mouse comunicar-se com um micro sem um cabo de conexão?
2) Qual a diferença entre um joystick e um joypad?
3) Cite os principais sistemas brasileiros de videotexto, ou bancos de dados de acesso público.
4) Os conceitos de "árvore de jogo", "nó" e "corte alfa" pertencem a qual área da ciência da computação?

RESPOSTAS DO TESTE ANTERIOR

1) A empresa estatal brasileira que transmite os bancos de dados de acesso público é a Embratel.
2) A linguagem C-LOGO, ou LOGO CONCORRENTE, tem os novos comandos FOREVER, para execução indefinida de uma rotina, e WHENEVER, para a execução condicional de uma função.
3) A linguagem ASSEMBLY apresenta-se em duas versões distintas, conforme o microprocessador empregado pelo micro: 6502 ou Z80.
4) O ASSEMBLY tem um comando equivalente ao GOSUB do BASIC: é o JSR, para o 6502, ou CALL, para o Z80.

PLANO DA OBRA

MICROCOMPUTADOR - CURSO PRÁTICO é uma coleção de 30 fascículos de periodicidade semanal. Os 29 primeiros terão, cada um, vinte páginas internas (miolo) e quatro capas, além de uma introdução de dezesseis páginas acompanhando o fascículo 1. O fascículo 30 incluirá o índice geral da obra. Os miolos, encadernados, constituem dois volumes de um curso prático de microcomputação, com seções indicadas por tarjas de diferentes cores.
As duas primeiras capas são descartáveis; as 3.as e 4.as trazem programas de jogos. Encadernadas, formam um volume de cem páginas.

COMO ENCADERNAR

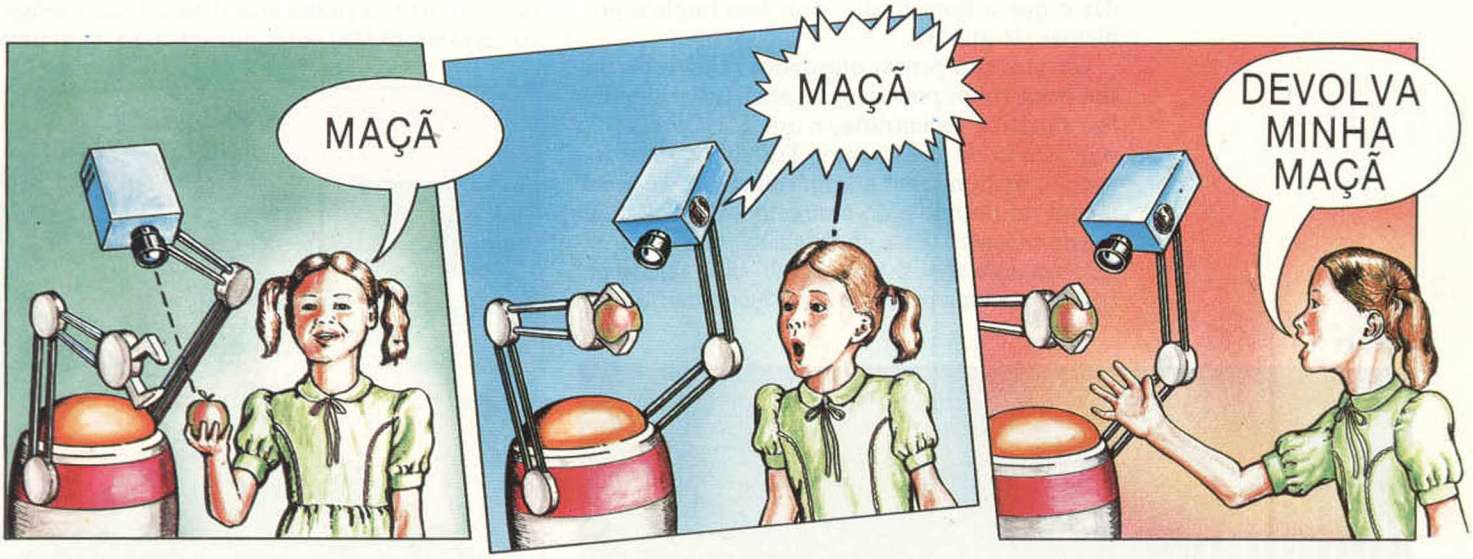
As capas duras, incluindo as guardas, estarão à venda simultaneamente com os fascículos 10 (**Volume 1**) e 30 (**Volume 2** e **Jogos**).
Volume 1 - Deve ser encadernado com os elementos dispostos nesta ordem: guardas; frontispício e introdução (publicados com o fascículo 1); miolos dos fascículos de 1 a 15; e guardas.
Volume 2 - Guardas; frontispício (que virá com o fascículo 30); miolos dos fascículos de 16 a 30; e guardas.
Jogos - Guardas; frontispício e sumário (que constituem as 3.as e 4.as capas do fascículo 30); 3.as e 4.as capas dos fascículos de 1 a 29; e guardas.

rioGráfica **CONSELHO DE ADMINISTRAÇÃO:** Roberto Irineu Marinho; João Roberto Marinho; José Roberto Marinho; Oscar Neves; **Diretoria:** Oscar Neves; Danilo Esteves Costa; João J. Noro; Marcos March; Nilo Sérgio de Almeida; **GRUPO EDUCAÇÃO E CULTURA - Divisão Comercial - Diretor:** Jaime Rodrigues; **Gerente de Planejamento:** João de Azeredo Coutinho; **Gerente de Vendas:** Rubens Barbosa; **Coordenadores de Promoção:** Paulo Cesar M. Seixas; Edgard Mello M. Neto; **Gerente de Circulação:** Norberto Martin; **Supervisor de Operações:** Abel Pereira Pinto; **Editorial - Diretor:** João J. Noro; **Editor-chefe:** Maurício Rittner; **Diretor de Arte:** Adalberto Cornavaca; **Administração Editorial:** Heitor de Souza Paixão; **Editor:** Jesse Navarro; **Editor-assistente:** Claudio Marcondes; **Coordenadora de Revisão:** Isis Augusta Loyolla; **Revisores:** Maria da Graça Mendonça Couto; Sérgio Roberto Torres; **Chefe de Arte:** Bonifácio Duardes Miranda; **Chefe de Estúdio:** José Yuji Kuribayashi; **Assistentes de Arte:** Miguel Luis Escámez Simón; Edemilson Ribeiro da Silva; Wagner Celestino; **Colaboradores - Editores-assistentes:** Isa Mara Lando; Luiz Carlos Pizarro Marin; **Texto:** Carlos Eduardo Silveira Matos; José Antônio Arantes; Jussara Maturo; Paulo Brito; Solange A. Menezes; **Preparação de Texto:** Luiz Carlos Cardoso; **Tradução:** Luiz Roberto de Godoy Vidal; Maria Clara Cescato; Maria Ester Menezes Martinho; Fernando Dorfman Knijnik; Newton Roberval Eichemberg; **Pesquisa:** Stella Maria Quentel; **Assessoria Técnica:** Geraldo Coen; Gustavo José Ferreira Grubba; José Rubens Salles Toledo; Lauro de Lauro; Pierluigi Piazzi; **Estúdio:** Vagner Facuri de Oliveira. As fotos não creditadas pertencem à obra original; Microsoft. © APSIF Copenhagen, 1984; © Orbis Publishing Ltd., 1984; © Editora Rio Gráfica Ltda., 1985, para a língua portuguesa. **Composição:** Editora Rio Gráfica Ltda. e AM Produções Gráficas Ltda. **Impressão:** JBIG.

NÚMEROS ATRASADOS - Você poderá comprar os exemplares que faltam em sua coleção, pelo preço do último fascículo posto à venda, em sua banca de jornal preferida. Caso não consiga obtê-los, faça sua solicitação por carta endereçada diretamente à Editora Rio Gráfica Ltda., rua Itapiru, 1209, Rio Comprido, Rio de Janeiro, CEP 20251. O atendimento será feito por via postal. Nas cidades do Rio de Janeiro e de São Paulo, as compras poderão ser efetuadas pessoalmente nos seguintes endereços: Rio de Janeiro - Rua Itapiru, 1209, Rio Comprido; São Paulo - Rua Frei Caneca, 1152, Consolação.

Distribuidor exclusivo para todo o Brasil: Fernando Chinaglia - Rua Teodoro da Silva, 907, Rio de Janeiro. Distribuidor para Portugal: Electroliber Lda. - Rua Prof. Reinaldo dos Santos, 1488, Lisboa.

CONVERSA DE ROBÔ



Dotar um robô de voz revela-se tarefa das mais difíceis, pois ainda não compreendemos a fundo o aprendizado da fala. Mas algumas teorias importantes podem esclarecer muitas questões nesse sentido.

O estudo da fala humana deu origem a duas linhas de pensamento: uma crê que a linguagem é inata; a outra, que é adquirida, ou aprendida. Os defensores da primeira hipótese afirmam ser o homem a única criatura a se comunicar por meio da linguagem; os da segunda citam experiências com animais ensinados para se comunicarem com os seres humanos por meio da linguagem de sinais.

Se de fato aprendemos a falar pela simples exposição à fala, justifica-se um método que leve os robôs ao mesmo resultado. Afinal, tudo ficaria mais fácil se ele aprendesse a linguagem apenas ouvindo-a em uso.

Houve algumas tentativas de expandir o conhecimento gramatical do computador através de exemplos adicionais de estruturas de sentenças; outras procuraram dar ao robô a capacidade de aprender novas palavras e morfemas (elementos da linguagem) em qualquer idioma pela simples exposição. Até hoje, porém, nenhum dos sistemas criados conseguiu ensinar um robô a falar.

Assim, para fins práticos, a capacidade de fala do robô depende da tese que afirma ser a linguagem inata, não aprendida, devendo-se elaborar as regras lingüísticas e incorporá-las permanentemente ao robô, tal como se ele tivesse nascido com elas. Em termos gerais, esse procedimento se divide em duas fases: as análises sintática e semântica.

A análise sintática ocupa-se da gramática daquilo que se diz e decodifica a estrutura superficial da mensagem ou codifica-a numa forma gramatical pronta para transmissão pelo robô. O método mais comum é o uso da "árvore de análise", que decompõe ou compõe uma sentença a partir das várias partes da fala. Embora não seja fácil, essa tarefa vem se desenvolvendo satisfatoriamente.

A análise semântica oferece mais problemas, por envolver a elaboração do sentido da mensagem (quando o robô ouve alguém falar) ou a criação da mensagem que deve ser transmitida (quando ele quer falar com alguém). Ocorre que o significado depende do contexto em que a linguagem é falada, e isso também se aplica a todo o contexto da mensagem. Esse contexto abrange tanto o conhecimento sobre o estado do mundo enquanto se fala quanto o conhecimento que cada um dos interlocutores tem do outro.

Essa abordagem foi adotada em experiências realizadas pelo cientista em computação Terry Winograd, criador de um programa que possibilitou a um robô compreender o conteúdo da fala e agir sob instruções. Mas Winograd simulou em computador um robô que funcionava num mundo minuciosamente definido — no caso, alguns elementos básicos manipuláveis. O programa de Winograd, chamado SHRDLU, alcançou uma boa análise semântica, mas o mundo que ele podia entender era extremamente sim-

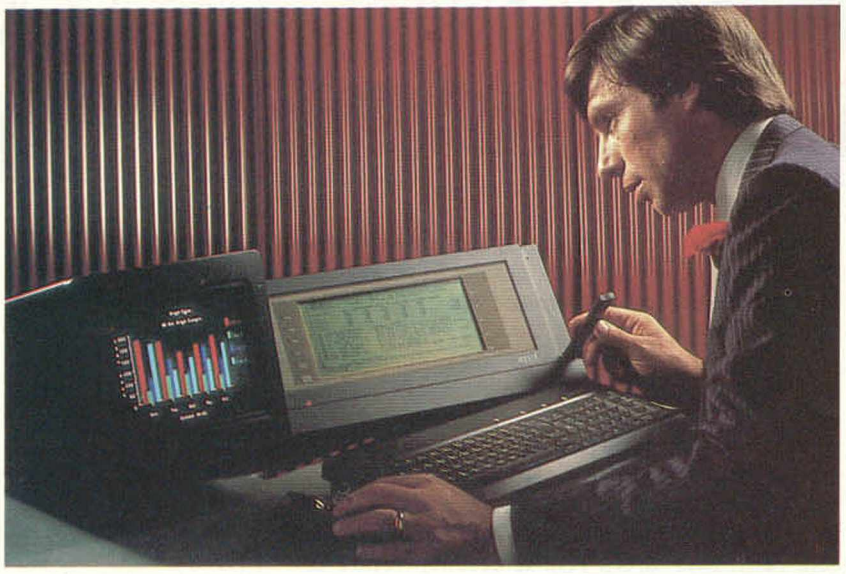
Ver para crer
Ao ver um objeto e lhe dar um nome — maçã, por exemplo —, o homem compreende o significado de "maçã". O robô, por sua vez, é capaz de reconhecer o objeto (fazendo-o corresponder a uma imagem interna) e de repetir o padrão sonoro que combina com maçã; mas não sabe que o objeto é uma fruta comestível e que, de fato, "pertence" ao homem — coisa que o ser humano compreende perfeitamente.

ples. Um robô que trabalhasse no caos do mundo real encontraria dificuldade bem maior em compreender o conteúdo da fala.

O uso proveitoso da linguagem pelos robôs exige uma mensagem entre o usuário e a máquina, e vice-versa. Para o robô é relativamente fácil falar, pois tudo o que ele venha a expressar encontra-se confinado nos restritos limites de seu conhecimento. A dificuldade está em compreender o que o homem diz, pois isso implica problemas de análise.

Chegou-se a pensar que dados falados recebidos pelos robôs poderiam ser analisados por análise sintática da entrada, e que isso revelaria o significado da mensagem. Trabalhos recentes, porém, demonstram a importância do conhecimento do mundo e do contexto em que a mensagem se insere. Essa abordagem conduziu a experiências de análise sintática do sinal falado para obter-se uma primeira suposição quanto a seu significado. Assim, pelo conhecimento do mundo e das coisas que venham a ser expressas, o robô repassa a análise sintática inicial com o propósito de obter aos poucos uma análise correta do que se expressou. Mas isso ainda está muito além da capacidade dos robôs.

Apricot F1



Ouvir e falar
É relativamente fácil criar a fala do robô e do computador. O mercado oferece sintetizadores, como o Currah, até mesmo para uso nos micros de menor porte. Mas existem certas dificuldades de reconhecimento devido às variações no modo como os seres humanos pronunciam os sons das vogais e à capacidade de processamento e de memória necessária para manipular um vocabulário extenso. Sistemas como o Big Ears e o Apricot F1 dispõem de uma pequena série de comandos reconhecíveis incorporados, porém insuficientes para abranger mais que um número mínimo de operações.

A síntese da fala

O método mais simples de síntese de fala utiliza um gravador cassete, por meio do qual o robô reproduz uma mensagem humana. Talvez isso frustre um pouco a idéia que se tem da fala de um robô, mas é o ponto de partida de todos os sistemas de síntese de fala.

Vejamos quais as limitações desse método e de que modo se pode desenvolvê-lo. A limitação mais evidente está nas características de um gravador: é mecânico, caro, volumoso e sujeito a freqüentes avarias. Considerado isso, o próximo passo será converter a mesma mensagem à forma digital, de modo que possa ser armazenada num chip da memória do robô. Para tanto, usa-se um conversor analógico/digital, no qual se empregam números para representar a forma de onda da fala, que está em contínua variação. É exatamente esse método que se usa na gravação digital de música — por exemplo, nos sistemas de discos compactos.

Esse método também apresenta inconvenientes. Um deles é que o sinal digitalizado ocupa muito espaço na memória. A gravação com disco compacto examina amostras do sinal acústico cerca de 44.000 vezes por segundo, com uma resolução em torno de 16 bits (ou seja, a amplitude da forma de onda é armazenada como um número de 16 bits, possibilitando a distinção de 2^{16} níveis, em que $2^{16} = 65.536$). Com esse sistema, cada segundo de gravação ocuparia 88.000 bytes de memória. Fica claro que uma mensagem falada excede a capacidade de armazenamento de qualquer computador. Essa taxa de amostragem, entretanto, é aplicável apenas à reprodução de sons de alta fidelidade. Um sistema simples poderia funcionar com uma resolução de 8 bits e uma taxa de 3.000 amostras por segundo, exigindo apenas 3 Kbytes de memória.

Entretanto, para liberar o máximo de espaço de memória, outras economias são necessárias. A lingüística descobriu que a linguagem falada pode ser convenientemente decomposta em unidades de fala chamadas fonemas. De modo geral, concorda-se que no total existem quarenta diferentes fonemas na maioria das línguas faladas, sendo possível, assim, armazenar informações acústicas exatas, indispensáveis para descre-

ver cada um desses quarenta fonemas e utilizá-los como base para a fala do robô. As informações de fonemas em geral são mantidas com um chip sintetizador de fala fabricado comercialmente, cabendo ao robô apenas encadear tais fonemas para gerar a mensagem necessária. Esta geralmente é mantida numa cadeia de números de fonemas na memória do computador.

Para se programar a maioria dos sintetizadores de fala em uso, escreve-se a mensagem que o robô deve transmitir numa versão fonética. Assim, a mensagem "O que é isso?" poderia ser escrita "U kié içu?", o suficiente para o chip sintetizador produzir a cadeia correta de sons. Essa não é, exatamente, a mesma representação utilizada pela lingüística na descrição dos fonemas, mas seria suficiente para o robô.

Percebe-se, neste ponto, que o robô deixou de utilizar uma mensagem pré-gravada, passando a gerar sua própria mensagem. Graças a isso, dirá tudo o que lhe foi pedido sem necessidade de armazenar previamente toda a mensagem.

Assim, podemos tentar programar algumas regras gramaticais para que o robô diga algo inteiramente original. Com uma ressalva: o número de coisas diferentes que o robô quererá dizer será bastante limitado, não havendo necessidade de complicação — a menos que seja grande a curiosidade de ver os resultados possíveis.

Quem já ouviu alguma vez um sintetizador de voz no terminal de serviços de bancos mais modernos sabe que a qualidade da fala, embora compreensível, não é perfeita. Isso ocorre por dois fatores: primeiro, a inflexão de um fonema usado pelo homem varia de acordo com os fonemas anteriores e posteriores; segundo, o som geral da fala humana modifica-se conforme o significado pretendido. Por exemplo: "Quer fazer o favor de sentar-se?" e "Quer fazer o favor de sentar-se?" são duas mensagens escritas de forma idêntica, mas, faladas, terão efeitos bem diferentes. Imagine a primeira emitida por uma gentil anfitriã a seu convidado, e a segunda por um irritado professor a um aluno. Tentou-se captar tais entonações nos sistemas de síntese de fala, mas existem diferenças na aplicação, visto que o robô não conhece o significado das palavras que usa.

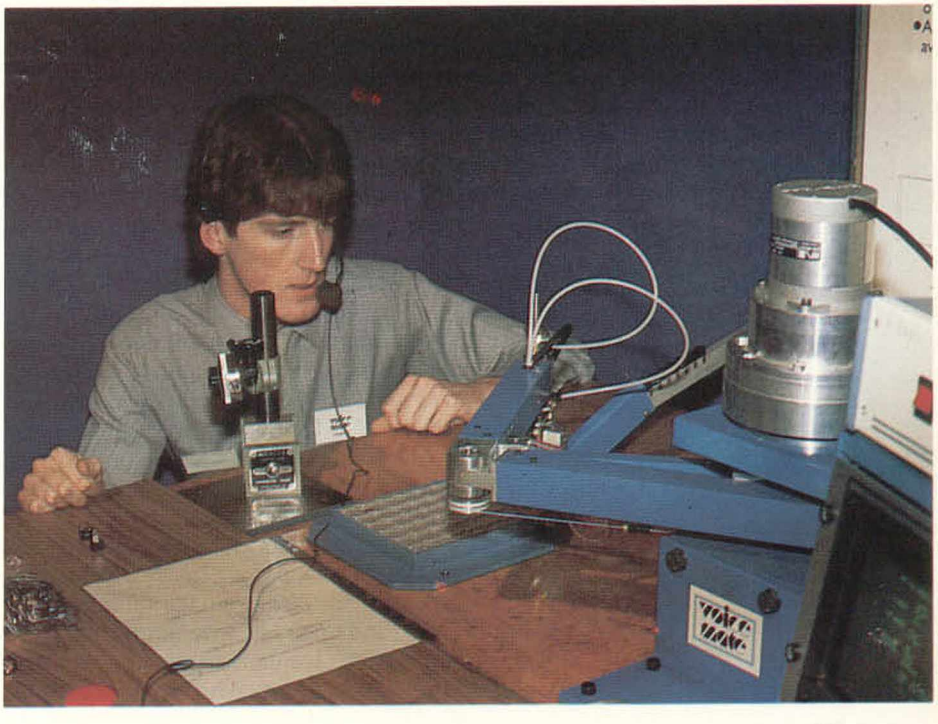
Reconhecimento

Ao se criar um sistema de reconhecimento de fala, um problema fundamental e que exige solução refere-se às coisas que se pode dizer a um robô e às inúmeras maneiras de exprimi-las. Pode-se abordá-lo inicialmente com o uso de gravação em fita daquilo que se pretende que o computador reconheça. Então, enquanto se fala, ele simplesmente examina todas as fitas gravadas em busca da que apresenta maior semelhança com a mensagem ouvida — é assim que muitos dos robôs reconhecem a fala. Armazenam alguns "gabaritos" internos de mensagens faladas e, quando se "conversa" com eles, simplesmente buscam o gabarito que melhor corresponda à mensagem recebida. Obtêm-se tais gabaritos treinando o robô, repetindo-lhe palavras ou frases, até que ele extraia uma "média" do que "ouviu". Esse método dá bons resultados quando a quantidade de coisas a dizer ao robô é pequena e elas são sempre ditas da mesma maneira ou quase — isto é, com robôs que respondem a comandos simples como "Para a frente", "Vire à esquerda", e assim por diante.

Esse, porém, é um problema relativamente simples, identificado como reconhecimento discreto da fala — cada item emitido é "discreto", ou seja, separado das outras mensagens por uma pequena pausa, durante a qual ocorre o silêncio. O verdadeiro problema surge quando se deseja que o robô faça uso de fala contínua, a fala normalmente usada pelos seres humanos. É interessante experimentar algo assim como "O poço secou". O resultado será mais ou menos "Upossussekô", com a mistura dos sons e palavras.

Ao ouvir falas alheias, as pessoas resolvem esse problema tentando supor o que o falante quer dizer (em geral uma tarefa não muito difícil) e utilizando a suposição para decodificar a mensagem. Para agir assim, porém, o robô teria de conhecer muito sobre a intenção e o significado possíveis — esta sim uma tarefa complexa.

De modo geral, a síntese da voz pelos robôs vem se tornando comum, embora ainda haja muito a fazer no sentido de melhorar a qualidade da fala. Árdua é a tarefa do aperfeiçoamento do reconhecimento da voz; e, quanto a isso, o que se conseguiu foi munir o robô de uma compreensão da fala equivalente à de um cão bem treinado, capaz de responder a comandos verbais, desde que pouco numerosos. Constata-se, porém, o enorme interesse na solução dos problemas relacionados à fala do robô, indicando, para um futuro próximo, consideráveis avanços.



Ordem sonora
O Voicemate, um braço mecânico controlado por voz, foi desenvolvido pelo Departamento de Ciências e Engenharia da Newcastle Polytechnich, para uso industrial e em laboratórios.

Antecipação
No início das pesquisas sobre inteligência artificial, a capacidade de um computador jogar xadrez era considerada a medida mais expressiva de sua potencialidade. Hoje, a maioria dos programas de xadrez possui recursos para antecipar uma série de movimentos do jogo, avaliando assim qual o melhor dos próximos lances. Para fazer isso, os computadores constroem uma árvore de busca.

GRANDE MESTRE

Os conceitos básicos por trás dos programas de xadrez geralmente envolvem "árvores" de busca. Vejamos alguns mecanismos que podem antecipar e determinar os melhores movimentos em jogos de estratégia.

As primeiras aplicações da inteligência artificial destinavam-se à resolução de problemas abstratos de matemática e de física. Paralelamente, porém, os pesquisadores — inclusive alguns pioneiros da computação, entre eles Claude Shannon, John von Neumann e Alan Turing — dedicaram-se à programação de máquinas para o xadrez, o jogo intelectual por excelência. Um programa bem-sucedido era considerado o teste supremo de inteligência de um computador.

Hoje existem sistemas computacionais com nível de jogo equivalente ao dos mestres internacionais — embora poucos afirmem que essas máquinas "pensem". Mesmo assim, o xadrez e outros jogos que exigem habilidade mental fornecem o terreno ideal para testar teorias de planejamento estratégico.

A maioria dos programas de jogos de habilidade apóia-se em técnicas de busca "em árvore", modificadas para levar em conta um adversário. A idéia fundamental é a de "antecipação". O programa desenvolve uma árvore de jogo considerando seus próprios movimentos, prevendo os possíveis contramovimentos do oponente, examinando antecipadamente suas respostas a eles e assim por diante.

A ilustração da página seguinte mostra a árvore de antecipação de um jogo imaginário entre MINI e MAX. A raiz da árvore é a posição atual, com MAX pronto para se mover. Os nós terminais, ou "folhas", são posições de final de jogo. Emprega-se a árvore para selecionar um movimento por meio de um processo conhecido como "minimaxização", claramente definido pela primeira vez em 1949, por Claude Shannon. De início, atribuem-se valores numéricos aos nós terminais — digamos, 1 para vitória, 0 para empate e -1 para derrota. Esses valores combinam-se à medida que avançamos na árvore, na suposição de que o jogador (MAX) sempre escolhe o maior valor, enquanto o oponente (MINI) sempre escolhe o menor, produzindo assim valores para os nós seguintes.

Neste exemplo, o valor da raiz é 0: o jogo deverá empatar, desde que ninguém cometa um erro. O movimento correto a esse nível será então M1, M3 ou M4, mas não M2. As regras que regem a ramificação e a geração de valores dos nós são determinadas pelas regras do jogo específico. Nos mais simples, como o jogo-da-velha, é possível apresentar a árvore do início até o final. Já o xadrez tem um "fator de ramificação" de cerca de 32; isso significa que existem aproximadamente 32 jogadas legítimas a partir de qualquer posição. Uma jogada antecipada em quatro lances (dois movimentos de cada lado) levaria a mais de um milhão de nós terminais. Essa explosão combinatória implica que programas de xadrez não podem ter seu desenrolar antecipado até o final da partida.

Em vez disso, esses programas costumam antecipar até onde podem e avaliar as posições aí encontradas. Tal procedimento exige método para decidir se os nós são favoráveis ou desfavoráveis; é a chamada avaliação estática, fonte inevitável de imprecisão, visto ser apenas uma estimativa do resultado final. A lógica do exame com certa antecipação, usando uma função de avaliação imperfeita, está na sua aplicação próximo ao final do jogo — e, de qualquer modo, será melhor que a ausência de verificação.

Num jogo de damas, por exemplo, poderíamos criar uma função de avaliação muito simples, com apenas quatro termos, baseada em:

D, vantagem de dama;
P, vantagem de peças;
M, diferença de mobilidade; e
C, controle do centro do tabuleiro.

Tais atributos podem ser calculados pelo exame do tabuleiro. Por exemplo, D = Dd-Do, sendo Dd as damas da defesa e Do, as damas oponentes.

As outras características refletem vários aspectos estratégicos do jogo. É melhor ter mais peças que o oponente (o perdedor termina sem peças); será útil ter mais movimentos disponíveis; as casas centrais possuem maior valor que as laterais. O programa deve, de alguma forma, combinar essas quantidades numa contagem geral.

Vamos estabelecer, por exemplo, que uma dama (D) vale três peças comuns (P), uma peça vale dois e meio movimentos adicionais (M) e um movimento simples vale duas vezes o controle de uma casa central (C). Nossa função de cálculo será:

$$V = 15D + 5P + 2M + C$$

(São usados pesos inteiros para acelerar o cálculo.)

Essa, entretanto, é uma função de cálculo muito rudimentar: o programa clássico de jogo de damas de Arthur Samuel, criado no início dos anos 60, empregava até 25 parâmetros. Os coeficientes neste exemplo também são bastante arbitrários. Um dos aspectos atraentes do programa de Samuel era que ele ajustava seus próprios pesos automaticamente, o que consistia numa forma rudimentar de aprendizado.

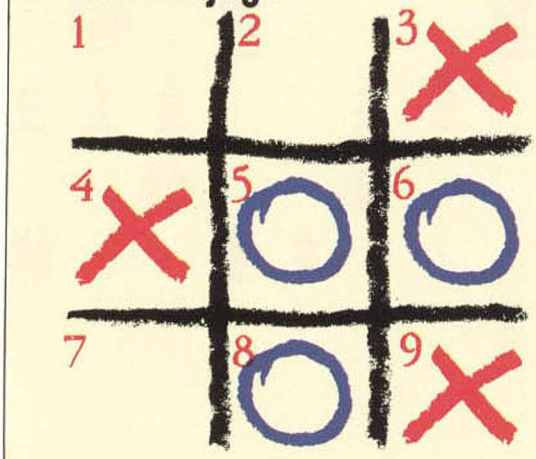
A idéia de atribuir valores numéricos a características de jogos e combiná-los numa soma ponderada para calcular o valor de qualquer posição vem mostrando sua importância desde a década de 50. A função de avaliação tem um papel semelhante ao da estimativa heurística da distância na solução de problemas por meio de busca.

Um programa que se limite a antecipar sempre e com a mesma profundidade acaba em situações difíceis. Isso porque algumas posições do jogo são "estáveis", enquanto outras são bastante "instáveis". No xadrez, é provável que o estado do jogo após a captura ou promoção de um peão seja instável ao extremo, pois uma recaptura pode ocorrer no próximo lance. Caso isso aconteça, num lance além do "horizonte" do programa, a avaliação estará seriamente comprometida.

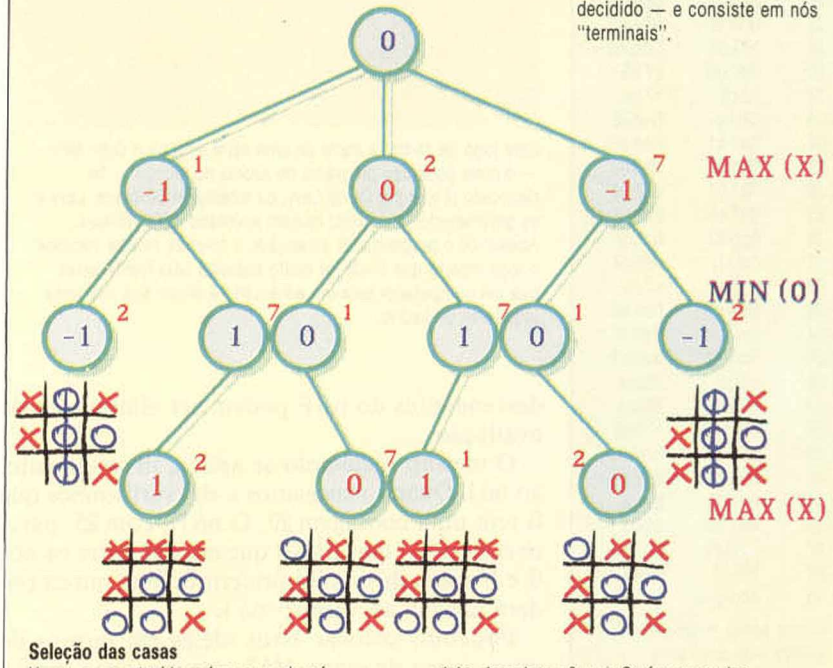
Para atenuar esse efeito, a maioria dos programas não possui antecipação com profundidade fixa, mas sim uma medida da "estabilidade" que indica se determinada posição pode ser avaliada de modo confiável. Se for evidenciada instabilidade na posição, a pesquisa avança ainda mais. Nos jogos de xadrez e damas, isso acarreta o exame comparativo de longas seqüências de captura.

O algoritmo alfa-beta surgiu em 1967, no programa MacHack, de Greenblatt. Trata-se de uma sofisticação da minimaxização básica — fornece o mesmo resultado, mas com esforço muito menor. O diagrama da página seguinte mostra parte da árvore de jogo entre MINI e MAX.

Estado do jogo



O diagrama mostra a situação num jogo-da-velha após seis lances (o xis joga em seguida). Pode-se construir uma árvore simples para acompanhar os próximos três movimentos finais, pelo exame das opções abertas a cada estágio.
A primeira jogada possui três nós, referentes às casas 1, 2 ou 7, que podem ser ocupadas pelo xis. Quando o círculo joga, há somente duas casas para escolha, com três possibilidades (duas das quais estarão disponíveis após o xis ter jogado). O próximo lance, portanto, tem seis nós.
O final indica situações em que todas as casas foram ocupadas — caso o jogo não tenha se decidido — e consiste em nós "terminais".



Seleção das casas
Uma vez construída a árvore, cada nó terminal recebe um valor: 1 para a vitória dos xis, 0 para um empate e -1 para a vitória dos círculos. Podemos então percorrer a árvore em sentido contrário, atribuindo valores aos nós anteriores. Se tomarmos o nó mais à direita na primeira jogada, chegaremos ao valor -1 considerando os dois nós abaixo dele, de valores 0 e -1. Se for a vez dos círculos, o valor mínimo será escolhido, isto é, -1. Voltando pela árvore até a jogada atual, os xis deveriam escolher o maior valor disponível. Neste exemplo, o melhor que podem fazer é empatar a partida, pela escolha da casa 2 no próximo lance; as outras opções dariam a vitória aos círculos.

Os números dentro dos nós são avaliações. As letras em cada nó (de A até L) mostram a ordem em que se examina a árvore, a partir de um procedimento de primeira profundidade. A barra simples marca os cortes alfa e as barras duplas os cortes beta. Estes eliminam as ramificações que não podem afetar o resultado final.

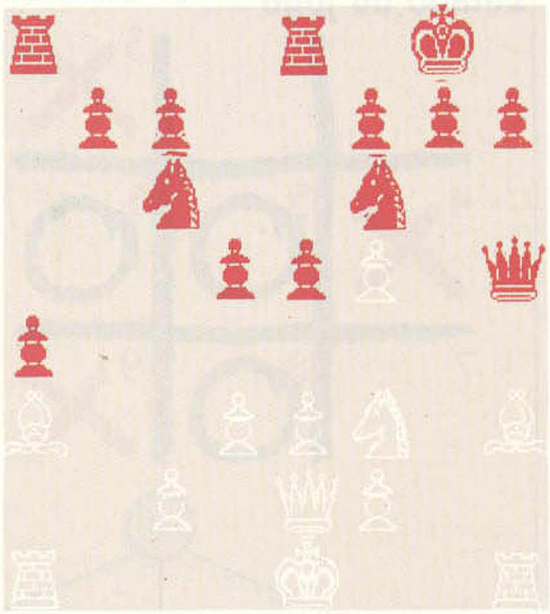
O corte alfa na posição E significa que esse nó não precisa ser avaliado, bem como seus eventuais descendentes. Quando alcançamos o nó E, sabemos que o C obtém contagem 15; mas, na posição D, o oponente pode forçar-nos a descer para 10. Inútil verificar se há possibilidade de sermos forçados ainda mais para baixo, pois essa rota é evidentemente menos desejável que aquela que passa pelo nó C. Assim, os outros

Movimentos

1	d2-d3	e7-e5
2	Nb1-d2	Nb8-c6
3	g2-g3	Ng8-f6
4	Bf1-g2	Bf8-c5
5	e2-e3	d7-d5
6	Ng1-e2	0-0
7	a2-a3	Bc8-f5
8	b2-b3	Nf6-g4?!
9	h2-h3	ng4-f6
10	Bc1-b2	Qd8-d6
11	g3-g4	Bf5-e6
12	Ne2-g3	a7-a5
13	Qd1-e2	Nf6-d7
14	Ng3-f5	Be6xf5
15	g4xf5	Nd7-f6
16	h3-h4	Rf8-e8
17	Bg2-h3	a5-a4
18	b3-b4	Bc5xb4!?
19	a3xb4	Qd6xb4
20	Bb2-a3	Qb4xh4
21	Nd2-f3	Qh4-h5
22	Nf3-d2	Qh5xe2+?
23	Ke1xe2	b7-b5
24	c2-c3	h7-h6
25	Bh3-g2	Ra8-a5
26	Ra1-b1	Re8-b8
27	Rb1-b2	Rb8-b6
28	Rh1-b1	Kg8-h7
29	Ba3-c5	Rb6-b8
30	Bc5-a3	Nc6-a7
31	Nd2-f3	Nf6-d7
32	Nf3-e1	c7-c6
33	Nd1-c2	Rb8-a8
34	Nc2-b4	Ra8-d8
35	Nb4-a2	Na7-c8
36	c3-c4!	d5xc4
37	d3xc4	b5xc4
38	Bg2xc6	Nc8-a7
39	Bc6-e4	Nd7-f6
40	Ba3-e7	Rd8-c8
41	Be7xf6	g7xf6
42	Rb1-b6!	c4-c3
43	Rb6xf6	Kh7-g7
44	Rf6-b6	a4-a3
45	Rb1-g1+	

Nesse ponto, o programa estava indicando uma desvantagem de 2,4 peões, e os programadores optaram por abandonar a partida.

Posição das peças depois do lance 21



Este jogo de xadrez é parte de uma série em que o Cray Blitz — o mais poderoso programa de xadrez do mundo — foi derrotado (4 a 0) por David Levy, da Intelligent Software. Levy e os programadores do Blitz haviam apostado 5.000 dólares. Apesar de o programa ter alcançado o nível de mestre nacional, o jogo mostra que ainda há muito trabalho pela frente antes que um computador seja um adversário à altura dos melhores jogadores de xadrez.

descendentes do nó F podem ser eliminados da avaliação.

O mesmo raciocínio se aplica, inversamente, ao nó I. Quando chegamos a ele, verificamos que G tem uma contagem 20. O nó H, com 25, parece melhor — mas é MINI que escolhe entre os nós G e J, e sem dúvida irá preferir G. MAX nunca poderá chegar ao valioso nó I.

Podemos colocar essas idéias em termos de uma guerra de sexos. MAX, um machista, pensa ser o nó C, por exemplo, tio dos nós D e E, ambos filhos de F. A feminista MINI, por seu lado, acha que G é a tia das irmãs H e I, filhas da mãe J. Se você não se importa com o fato de os nós mudarem de sexo em lances alternados, essa analogia o ajudará a compreender melhor a regra alfa-beta:

- Tão logo MAX encontre um filho menos importante que qualquer de seus tios, ignorará os demais irmãos desse filho.
- Tão logo MINI encontre uma filha menos importante que suas tias, ignorará as demais irmãs dessa filha.

No melhor dos casos, o algoritmo alfa-beta examina apenas duas vezes a raiz quadrada do número de nós terminais na árvore do jogo, ao contrário da minimaxização simples. No pior, examina a mesma quantidade de vezes, porém um pouco mais devagar. Para evitar o primeiro dos nossos dois casos, é importante gerar, numa ordem coerente, os irmãos e irmãs a cada nível. Nos níveis maximizantes, devem ser gerados primeiro os melhores nós; nos minimizantes, primeiro os piores (que são os melhores para o oponente).

O jogo de números

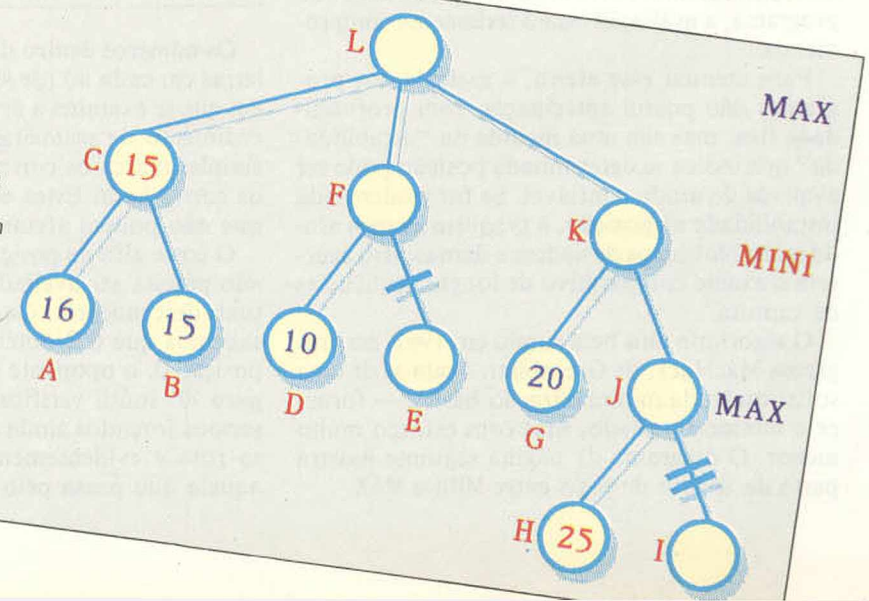
Para ilustrar os conceitos mais importantes da pesquisa em árvore, apresentamos um jogo artificial que abrange técnicas de pesquisa quase pura. Isso significa que os detalhes da representação no tabuleiro, geração de movimentos e avaliação estática — fundamentais em qualquer programa de jogo verdadeiro, mas específicos ao jogo em pauta — não mascaram a simplicidade básica do procedimento alfa-beta.

Não há tabuleiro nem peças: o estado do jogo é descrito por um único número, representado por V%. O computador deve elevar o valor de V% para 255; você deve torná-lo menor que 255.

A cada vez, o jogador pode utilizar uma entre quatro funções (A, B, C, D) referidas nas linhas de 1.030 a 1.060 do programa. Você pode

Podando a árvore

Ao eliminar ramificações redundantes da árvore, a "poda" alfa-beta melhora o método básico de minimaxização. Os cortes alfa (indicados por uma barra simples) se formam quando um percurso mínimo já foi encontrado na jogada de MINI, tornando desnecessárias buscas complementares. Os cortes beta (indicados pelas barras duplas) se formam quando é encontrado um caminho máximo com a jogada de MAX.



alterá-las para criar diferentes versões do jogo, tornando-o, por exemplo, mais difícil para o computador.

O jogo, embora simples, ilustra a estratégia da pesquisa, excluindo os detalhes relacionados a um jogo específico como o xadrez. Além disso, é altamente matemático — o computador leva vantagem. Nessa versão, a minimaxização alfa-beta apóia-se no uso de funções recorrentes com parâmetros e variáveis locais. Os parâmetros são os seguintes:

VV%, estado atual do jogo;
A%, melhor valor de alfa até agora nesse nível;
B%, pior valor de beta até agora nesse nível; e
D%, indicador de profundidade.

A versão do programa que fornecemos a seguir serve, praticamente sem alterações, para micros das linhas Apple, TRS-80 e Sinclair.

O jogo dos números

```
70  REM  ***JOGO DOS NUMEROS***
80  GOSUB 1000: REM  **INICIALIZACAO**
90  GOSUB 1600: REM  **INSTRUCOES**
100 :
110  REM  ***LOOP PRINCIPAL***
120  GOSUB 2000: REM  PREPARAR NOVO JOGO
130  INPUT "QUEM JOGA PRIMEIRO(1=VOCE,2=EU)";H1
140  IF H1 < 1 OR H1 > 2 THEN 130
150  REM  **LOOP DO JOGO***
160  IF H1 = 1 THEN  GOSUB 3000
170  REM  **VEZ DA PESSOA**
180  GOSUB 3500: REM  MOSTRA O TABULEIRO
190 H1 = 1: REM  SEMPRE 1 DEPOIS DO PRIMEIRO CICLO
200  GOSUB 4000: REM  TESTE DA VITORIA
210  IF EG = 0 THEN  GOSUB 5000
220  REM  **VEZ DO COMPUTADOR**
230  GOSUB 3500: REM  MOSTRA O ESTADO DO JOGO
240  GOSUB 4000: REM  TESTE DE FIM DE JOGO
250  IF EG = 0 AND M <  = 33 THEN 150: REM  VOLTA AO LOOP
260  REM  **FINAL***
270  GOSUB 6000: REM  FINALIZACAO
280  INPUT "OUTRO JOGO?(1=SIM,2=NAO)";Y
290  IF Y < 1 OR Y > 2 THEN 280
300  IF Y = 1 THEN 110: REM  NOVO JOGO
310  PRINT "ATE LOGO E OBRIGADO PELO JOGO"
320  END
330 :
500  REM  **MAXIMIZAR**
510 D = D + 1:C1 = C1 + 1
520  IF D >  = MD OR  ABS (V(D)) > HI THEN A(D) = V(D):D = D - 1: RETURN

530  REM  APROFUNDA NA ARVORE
540 P(D) = 0
550  REM  LENDO A ARVORE
560 P(D) = P(D) + 1:H = P(D):V = V(D)
565  GOSUB 5500: REM  FAZ UMA JOGADA
570  IF D = 1 THEN  PRINT  CHR$ (64 + H);" = ";
580 D1 = D + 1
590 A(D1) = A(D):B(D1) = B(D):V(D1) = V
595  GOSUB 700: REM  CHAMA MINIMIZAR
600  IF B(D + 1) > AD THEN A(D) = B(D + 1):K(D) = P(D)
610  IF D = 1 THEN  PRINT B(D + 1);" ";
620  IF P(D) <  = 3 AND A(D) < B(D) THEN 550
625  REM  GUARDA O MELHOR ATE AGORA
630  IF D = 1 THEN BV = A(D):HH = K(D)
640 D = D - 1: RETURN
650 :
700  REM  **MINIMIZAR**
710 D = D + 1:C2 = C2 + 1
720  IF D >  = MD OR  ABS (V(D)) > HI THEN B(D) = V(D):D = D - 1: RETURN

730 P(D) = 0
740  REM  **ATRAVES DA ARVORE**
750 P(D) = P(D) + 1:H = P(D):V = V(D): GOSUB 5500: REM  FAZ UMA JOGADA
760 D1 = D + 1:A(D1) = A(D):B(D1) = B(D):V(D1) = V
770  GOSUB 500: REM  CHAMA MAXIMIZAR
780  IF A(D + 1) < B(D) THEN B(D) = A(D + 1)
790  IF P(D) <  = 3 AND B(D) > A(D) THEN 740
800 D = D - 1: RETURN
810 :
1000  REM  ***INICIALIZA**
1010 BL$ = " "
1020  REM  DEFINICAO DAS 4 FUNCOES
1030  DEF  FN A(X) = 2 * X - 7
1040  DEF  FN B(X) =  INT (X / 2) + 1
1050  DEF  FN C(X) = 4 * X + 17
1060  DEF  FN D(X) = 3 * X + 4
1070 LO =  - 255:HI = 255
1080  REM  MATRIZES USADAS PELO MINIMAX
1090 D = 16
1100  DIM V(D),A(D),B(D),P(D),K(D)
1110  RETURN
1120 :
1600  REM  INSTRUCOES
1610  PRINT "BENVINDO AO JOGO DOS NUMEROS"
1620  PRINT "EU TENTO MAXIMIZAR"
1630  PRINT "VOCE TEM QUE MINIMIZAR"
1640  PRINT "PARA VER O EFEITO DE UMA JOGADA"
1650  PRINT "DIDITE A,B,C OU D E EM SEGUIDA X"
1660  PRINT : RETURN
1670 :
2000  REM  **PREPARACAO**
2010 M = 0:V =  INT ( RND (1) * 15) - 8: REM  ESTADO INICIAL
2020 EG = 0
2030  PRINT "ESTADO INICIAL= ";V
2040  RETURN
2050 :
3000  REM  **JOGADA DA PESSOA**
3010 M = M + 1: PRINT
3020  PRINT "SUA JOGADA E ";
3030  INPUT H$
3040  IF H$ = "A" THEN  PRINT  FN A(V):H = 1
3050  IF H$ = "B" THEN  PRINT  FN B(V):H = 2
3060  IF H$ = "C" THEN  PRINT  FN C(V):H = 3
3070  IF H$ = "D" THEN  PRINT  FN D(V):H = 4
3080  IF H$ <  > "X" THEN 3020: REM  JOGADA NAO FOI SELECIONADA
3090  GOSUB 5500: REM  FAZ UMA JOGADA
3100  RETURN
3110 :
3500  REM  MOSTRA O TABULEIRO
3510  PRINT : PRINT "      JOGADA";M;" --)";
3520  IF M < 1 THEN  RETURN
3530  PRINT  CHR$ (64 + H);
3540  PRINT " = ";V: PRINT : RETURN
3550 :
4000  REM  TESTE DE VITORIA
4010  IF M < 1 THEN  RETURN
4020 EG = 0
4030  IF V < LO THEN EG =  - 1
4040  IF V > HI THEN EG = 1
4050  RETURN
4060 :
5000  REM  **JOGADA DO COMPUTADOR**
5010 W = V: REM  GUARDA O ESTADO ATUAL
5020 M = M + 1
5030 MD = 6: REM  MAX PROFUNDIDADE
5040  IF M < 4 THEN MD = 4
5050  IF M > 8 THEN MD = 8
5060  GOSUB 5200: REM  ---> H
5070 V = W: REM  VOLTA AO ESTADO INICIAL
5080  GOSUB 5500: REM  FAZ UMA JOGADA
5090  RETURN
5100 :
5200  REM  SELECIONA A JOGADA
5210 BV = LO:D = 0
5220 V(1) = V:A(1) = LO:B(1) = HI
5230  GOSUB 500: REM  MAXIMIZE
5240 H = HH
5245  PRINT
5250  INPUT "TECLE 1 PARA CONTINUAR";Q
5255  IF Q = 1 THEN 5260
5260  RETURN
5270 :
5500  REM  **FAZ UMA JOGADA**
5510  IF H = 1 THEN V =  FN A(V): RETURN
5520  IF H = 2 THEN V =  FN B(V): RETURN
5530  IF H = 3 THEN V =  FN C(V): RETURN
5540  IF H = 4 THEN V =  FN D(V): RETURN
5550 :
6000  REM  **FINALIZACAO**
6010  PRINT : PRINT "FIM DE JOGO"
6020  IF EG > 0 THEN  PRINT "EU GANHEI"
6030  IF EG < 0 THEN  PRINT "VOCE GANHOU,PARABENS"
6040  IF EG = 0 THEN  PRINT "JOGO EMPATADO"
6050  RETURN
6060 :
```

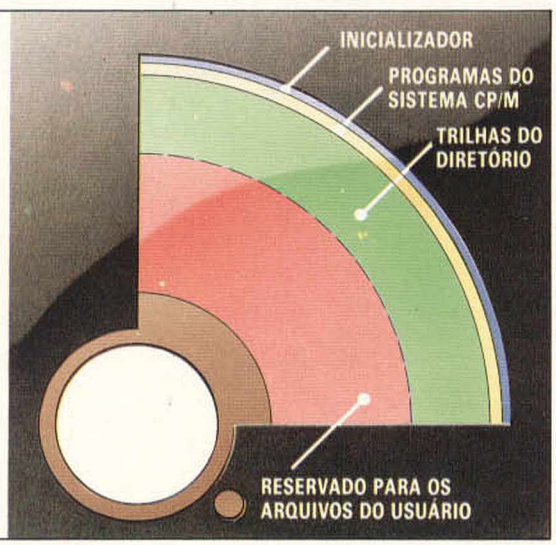
TRABALHO DE BASTIDOR

O que faz do CP/M o sistema operacional baseado em disco mais difundido em todo o mundo? O pioneirismo? A simplicidade e confiabilidade? Vejamos as respostas examinando como ele funciona.

Na primeira metade da década de 70, conquistas sucessivas no campo da informática criaram condições para a revolução dos microcomputadores. Em 1973, surgiu o primeiro microprocessador eficaz de 8 bits — o 8080, da Intel. No ano seguinte apareceu o disco flexível, capaz de agilizar e ampliar consideravelmente os recursos de memória e manipulação de dados computadorizados. Nesse mesmo período, um funcionário da Intel chamado Gary Kindall projetou o CP/M (Control Program/Microprocessors, "programa de controle para microprocessadores"), o primeiro sistema operacional baseado em discos flexíveis destinado a rodar em equipamentos de 8 bits. Em 1975, o CP/M recebia suas primeiras licenças comerciais.

Fazendo trilhas

A maior parte de um disco de dados CP/M está à disposição do usuário, mas as trilhas iniciais são reservadas ao próprio sistema. A trilha zero é a do inicializador, que carrega o CP/M de modo automático e reinicializa o sistema; as trilhas um e dois contêm o próprio CP/M. Seguem-se as trilhas do diretório, que contêm os Blocos de Controle de Arquivo, mostrando onde se localizam os registros de determinado arquivo.



A partir daí, os sistemas operacionais se multiplicaram. Todos, porém, atendem ao mesmo objetivo. Um sistema operacional é um programa em processamento permanente, encarregado de supervisionar as atividades do computador. Incumbe-se do controle de inúmeras tarefas intrínsecas ao funcionamento da máquina, desde formação de uma simples letra no vídeo até o gerenciamento de vários periféricos. Enquanto o sistema operacional atua "nos bastidores", o usuário pode dedicar a totalidade de sua atenção a um programa editor de texto, a uma planilha para cálculos ou a qualquer outro pacote de software que esteja rodando para atender a suas necessidades específicas.

Alguns sistemas operacionais são residentes em ROM — a memória permanente, inacessível ao operador. Por esse motivo, muitos usuários chegam a ignorar sua presença. Costuma-se encontrar esses sistemas fixos em estruturas de computação de menor porte e recursos igualmente fixos, enquanto os chamados DOS (Disk Operating System, "sistema operacional de disco"), carregados na RAM, incrementam a potencialidade do computador em que são executados.

Os DOS apresentam uma diversificação cada vez maior. No entanto, boa parte continua a se inspirar na lógica desenvolvida a partir de 1974 para o CP/M e que o transformou no DOS mais difundido em todo o mundo.

CP/M

Entre os fatores responsáveis pelo êxito do CP/M está sua simplicidade, que permite um acesso fácil e rápido aos arquivos de dados e programas. É, além disso, um sistema adaptável à vasta gama de computadores que utilizam chips 8080 ou compatíveis, o que implica a possibilidade de rodar enorme número de programas.

Quando um computador com CP/M é ligado, uma rotina residente em ROM aciona o Cold Start Loader ("carregador de partida a frio"), um pequeno programa destinado a carregar na memória RAM todo o sistema operacional. Na organização da memória, a área correspondente aos 256 primeiros bytes denomina-se página zero. Trata-se da zona de referência do sistema. Nela estão inscritos dados e indicadores situados em endereços fixos, imutáveis em qualquer sistema ou configuração.

Após a página zero vem a chamada zona TPA (Transient Program Area, "área de programas transitórios"). É a memória utilitária, de dimensões variáveis conforme a configuração do sistema, onde serão carregados os programas e comandos temporários do CP/M. Segue-se o CCP (Console Command Processor, "processador de comandos do terminal"), que analisa e executa os comandos emitidos pelo usuário.

Quando a zona TPA não basta para conter todo um programa em execução, o CCP pode ser "destruído" para criar-se mais espaço. Um RESET (reinicialização) no sistema aciona a rotina

Warm Start ("partida quente"), que acarreta uma expansão da memória TPA.

A parte subseqüente corresponde aos FDOS ("DOS funcional"), subdividido nas zonas BDOS ("DOS básico") e BIOS (Basic Input/Output System, "sistema básico de entrada e saída"). A zona BDOS consiste num conjunto de rotinas especiais para as operações de leitura/registro dos arquivos. Contém o sistema de gerenciamento dos arquivos em disco.

A zona BIOS depende do hardware onde é implantada. Consiste na única porção do CP/M que deve ser adaptada ao computador. Para isso, o fabricante de hardware recebe o programa padrão e reconfigura o trecho final em função do modelo de microcomputador em que o programa será executado.

Tal organização permite distinguir entre comandos residentes (permanentes na memória) e transitórios, carregados no disco e chamados à zona TPA caso necessário. Desse modo, ganha-se capacidade na RAM para as operações de processamento.

Entre os comandos residentes estão os de consulta ao diretório (índice) do disco, os de eliminação de organização e os de mudança de nome de arquivos. Alguns dos principais comandos relativos a periféricos dizem respeito à comunicação entre estes e os arquivos, bem como à possibilidade de se obterem cópias do próprio sistema operacional.

Apesar de sua confiabilidade e extraordinária popularização, o CP/M revelou, com o tempo, algumas debilidades. Em princípio deveria ser compatível com qualquer tipo de disco, impressora ou terminal, mas a experiência mostrou que não se devem usar discos com capacidade superior a 8 Mbytes.

Outra limitação diz respeito à memória: o CP/M exige pelo menos 20 Kbytes de RAM contínua, o que implica 64 Kbytes para poder operar. Em contrapartida, o TRSDOS, o NEWDOS e outros DOS em que o programa interpretador é residente em ROM exigem apenas 32 Kbytes de memória, embora sejam mais poderosos e mais lentos do que o CP/M.

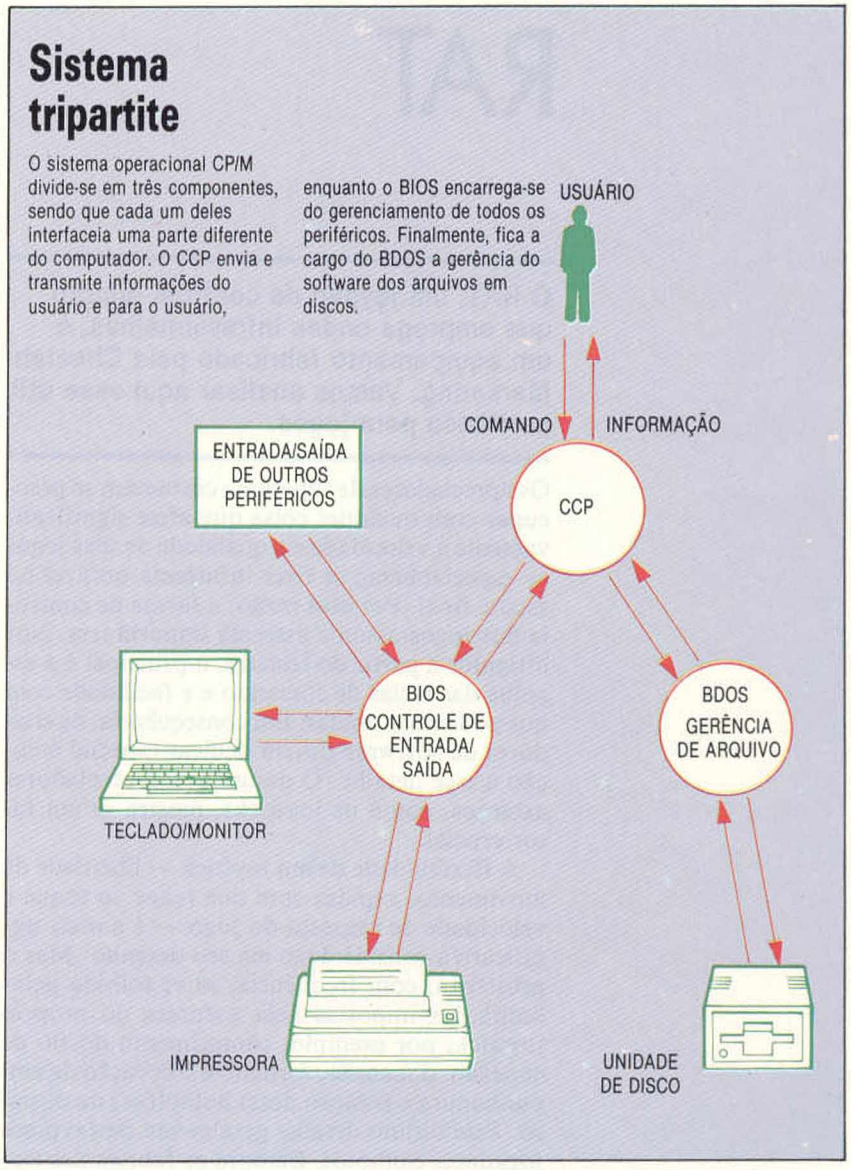
MP/M

Esse sistema (Multiprogram/Monitor, "multiprograma/monitor") representa um passo adiante em relação ao CP/M a nível de complexidade. Roda todos os programas para CP/M, mas é multiprogramável, podendo trabalhar com até dezesseis terminais (para dezesseis usuários). Os programas não se desenvolvem num lugar fixo na memória, podendo ser colocados em diferentes áreas e executados simultaneamente com outros programas em outras posições. Em outras palavras, o MP/M executa várias tarefas em aparente simultaneidade. De fato, o computador executa seqüencialmente fragmentos mais ou menos entrelaçados dessas diferentes tarefas.

Nesse sistema, cada terminal é um usuário e tem um diretório específico, podendo, entretanto, acessar arquivos do sistema central ou de outros usuários. Mas conseguem-se proteger os arquivos através de senhas.

Sistema tripartite

O sistema operacional CP/M divide-se em três componentes, sendo que cada um deles interfaceia uma parte diferente do computador. O CCP envia e transmite informações do usuário e para o usuário, enquanto o BIOS encarrega-se do gerenciamento de todos os periféricos. Finalmente, fica a cargo do BDOS a gerência do software dos arquivos em discos.



CP/NET

O CP/NET é um sistema operacional modular para redes, residente num microcomputador principal (em MP/M), em micros secundários (em CP/M) e em vários periféricos de grande porte, velozes e caros.

O desenvolvimento do CP/NET expressa uma preocupação racionalizadora a nível de custos: como utilizar equipamentos caros (os periféricos) sem onerar o preço de cada configuração? A resposta foi a montagem de uma rede, na qual o CP/M e o MP/M não são modificados. Em relação ao requisitante (o CP/NET), a rede controla as operações de entrada e saída, além de enviar mensagens destinadas a periféricos remotos; no servo (o MP/M), a rede consiste num conjunto de processos em andamento e que controla todas as mensagens dos requisitantes, executando as operações necessárias.

RAT

O RAT, um joypad de controle remoto que emprega ondas infravermelhas, é um equipamento fabricado pela Cheetah Marketing. Vamos analisar aqui esse útil periférico para jogos.

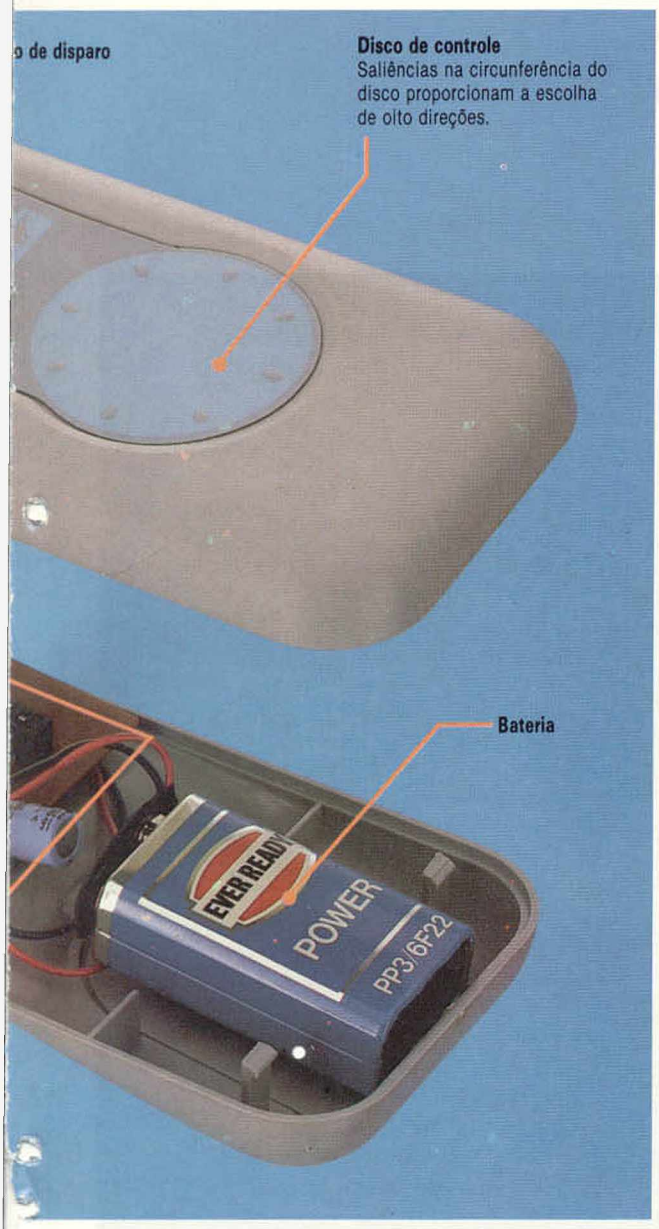
Os apreciadores de fliperama costumam se preocupar com qualquer coisa que afete significativamente a velocidade e a qualidade de seus jogos — especialmente se tiver influência notável no escore final. Por essa razão, a forma de controle dos jogos assume extrema importância. Nos dirigidos a partir do teclado, o principal é a escolha das teclas de comando e a facilidade com que podem ser usadas. Em conseqüência, os criadores de software devem dedicar especial atenção a esse detalhe. O design dos controladores externos, como os joysticks, mostra-se um fator crucial.

A flexibilidade de um joystick — liberdade de movimento, rapidez com que reage ao toque e velocidade de resposta do jogo — é a mais significativa consideração em seu desenho. Mas o projetista, com freqüência, se vê tolhido pelas limitações impostas pela natureza do próprio joystick; por exemplo, comprimento do fio de conexão, dimensão, formato e colocação da empunhadura e posição do(s) botão(ões) de disparo. Este último detalhe geralmente desfavorece jogadores canhotos. Embora os fabricantes tentem desenvolver projetos que superem algumas dessas desvantagens, nenhum obteve tanto sucesso como o joypad (tablete para jogos), comando remoto por infravermelho, da Cheetah Marketing, feito para o Sinclair Spectrum.

A Cheetah deu o nome de RAT (''rato'') ao seu joypad. Consta que essa sigla seria uma abreviação para Remote Action Transmitter (''transmissor de ação remota''), mas parece ter sido mesmo uma brincadeira com o sinônimo mouse (camundongo), aplicada aos controladores manuais usados com o Macintosh, da Apple, e outros micros. O RAT lembra uma arma phaser, levemente alongada, extraída do seriado de tevê *Jornada nas estrelas*. É longo, achatado, de cor cinza, com um disco azul de controle e um botão de disparo alaranjado. Dois transmissores infravermelhos projetam-se à frente da unidade. Quando se segura o RAT pela primeira vez, quase se espera que ele emita centelhas azuis.

O sistema também inclui sua própria interface, uma caixinha preta que se ajusta no conector edge, na parte posterior do Spectrum. Essa interface tem saída própria para expansões com-

Luz irradiante
A radiação infravermelha é produzida no transmissor por IEDs (Infrared Emitting Diodes, "diodos emissores de infravermelho") quando uma corrente elétrica atravessa um delgado chip de arsenieto de gálio, excitando suas moléculas e liberando fótons. No receptor, inversamente, uma corrente elétrica flui no IED quando a luz infravermelha incide no arsenieto de gálio. Ao se pressionarem os botões de controle do RAT, portanto, os dois IEDs transmissores emitem pulsos codificados de infravermelho em um feixe amplo, disparando o receptor diretamente, ou após a reflexão nas paredes do ambiente.



Recepção mista
O teclado original do fracassado PC Junior, da IBM, notabilizou-se principalmente pela pobreza de aspecto e configuração; mas esse micro teve o mérito de ser o primeiro com ligação infravermelha entre o teclado e o processador.

plementares. Um único receptor infravermelho na frente dessa unidade comunica-se com o RAT.

O pacote contém instruções que explicam como se usa o joypad e em quais jogos se pode aplicá-lo (qualquer software compatível com o joystick Kempston). Com grande visão, a Cheetah incluiu rotinas em BASIC e em código de máquina que possibilitam ao usuário incorporar o controlador RAT a seus próprios jogos.

As instruções afirmam que o RAT pode ser usado a distâncias de até 4 m apenas apontando-se "na direção do computador". O movimento é efetuado pressionando-se levemente o disco azul. Neste, oito pequenas saliências indicam, ao serem pressionadas diretamente ou em suas proximidades, a direção desejada — norte, sudeste, oeste etc., como numa bússola. Enquanto uma das mãos segura o RAT e controla a direção do movimento na tela, a outra comanda o botão de disparo. Devido ao desenho do RAT, não faz diferença qual das mãos realiza cada tarefa, pois o joypad ajusta-se bem tanto para destros como para canhotos. O transmissor requer uma bateria de 9 V, que se encaixa num pequeno alojamento, na parte posterior da unidade, diretamente abaixo do disco azul.

Uma vez conectada a caixa na interface do micro e carregado no Spectrum um jogo que requeira joystick, o usuário acha-se pronto para jogar. Como não há sinal visível de que o transmissor está ou não funcionando até se visualizarem os movimentos na tela, o usuário tende a posicionar-se tão próximo do computador quanto com um joystick comum. Existe certa relutância em aceitar-se uma distância de controle de 4 m; mas, quando se percebe que o RAT realmente funciona, tenta-se experimentá-lo ao máximo "para ver até onde ele vai".

De fato, o transmissor de ação remota funciona muito bem a distâncias até mesmo superiores a 4 m, e não precisa ser apontado em direção ao computador. O RAT atua até quando apontado para o teto, para o chão, para trás do usuário ou para os lados (embora seja um pouco difícil saber o que se está fazendo quando o transmissor é apontado em ângulos estranhos). O aparelho da Cheetah dá ao jogador enorme liberdade de movimentação. A maior desvantagem, entretanto, reside nas oito posições de movimento — acima, abaixo, direita, esquerda e os quatro pontos intermediários. Seria melhor se tivesse mais opções de controle.

O fato de não ter partes móveis torna o RAT uma unidade menos propensa a desgaste e ruptura do que os joysticks comuns, devendo ter uma vida útil bastante longa. De fato, a unidade vem com garantia de um ano. O transmissor da Cheetah custa somente um pouco mais que a maioria dos outros joysticks acrescidos da Interface 2 (o que não consola muito, se você já tem a Interface 2). Mas sua flexibilidade permite muito mais liberdade de movimento e um controle muito superior ao da maioria dos joysticks.

TELEDADOS

O mundo na tela
O videotexto coloca ao alcance do usuário bancos de dados com informações de praticamente todas as áreas do conhecimento humano. Até mesmo do exterior. Digitando palavras-chaves, o assinante vai selecionando os registros até chegar ao serviço que deseja. As informações podem ser, então, lidas, gravadas em disquete, em cassete ou impressas em papel.

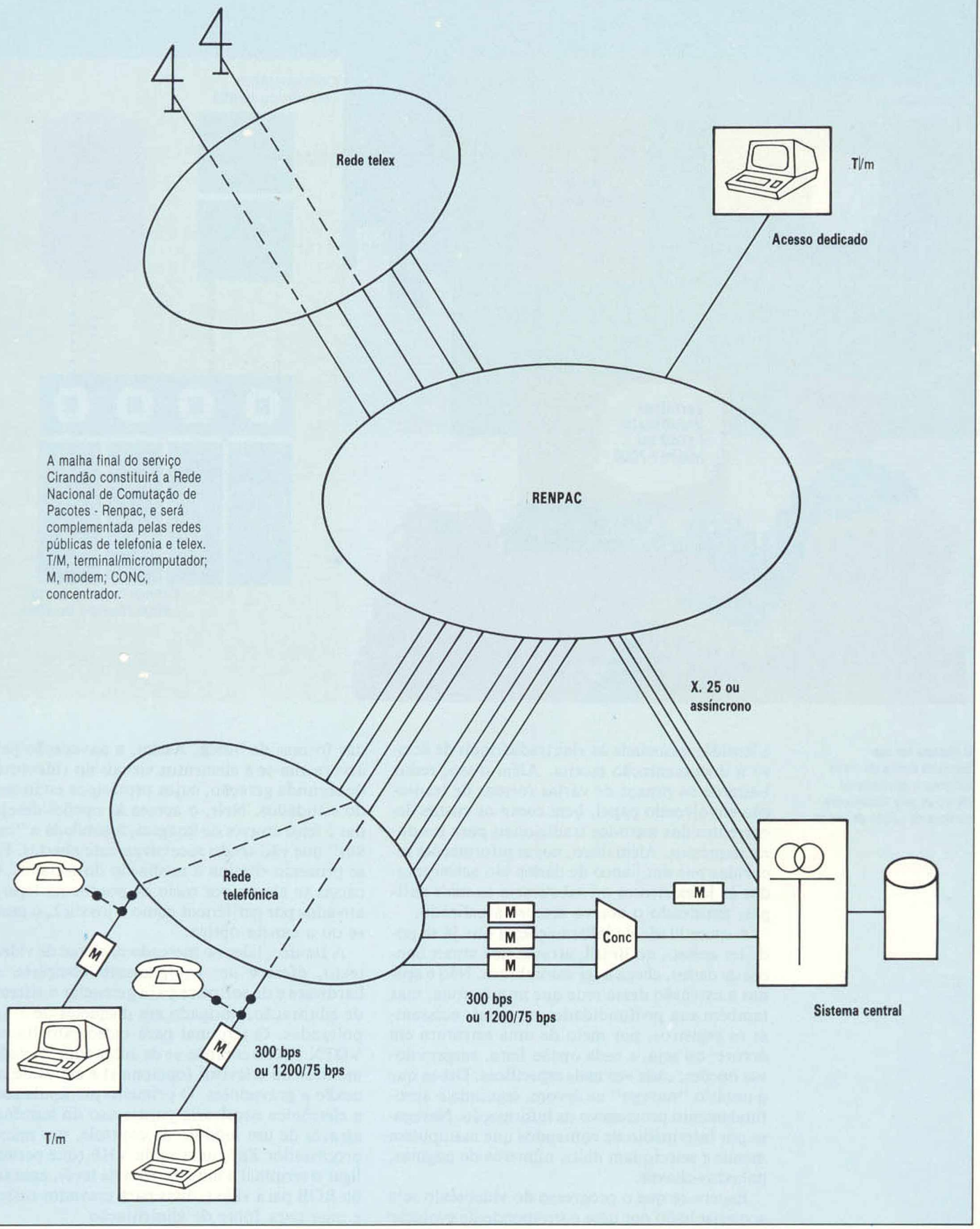
Talvez nenhum outro recurso tenha, como o videotexto, simplificado de modo tão acentuado o esforço pela busca da informação. O tempo se reduz a segundos e o espaço, à distância entre dedo e tecla.

O videotexto nasceu da aliança de dois dos meios de comunicação mais poderosos, o telefone e a televisão. Por meio de um terminal, acessam-se todas as informações não reservadas contidas num banco de dados de uma empresa pública ou privada, que são exibidas numa tela de televisão ou de microcomputador.

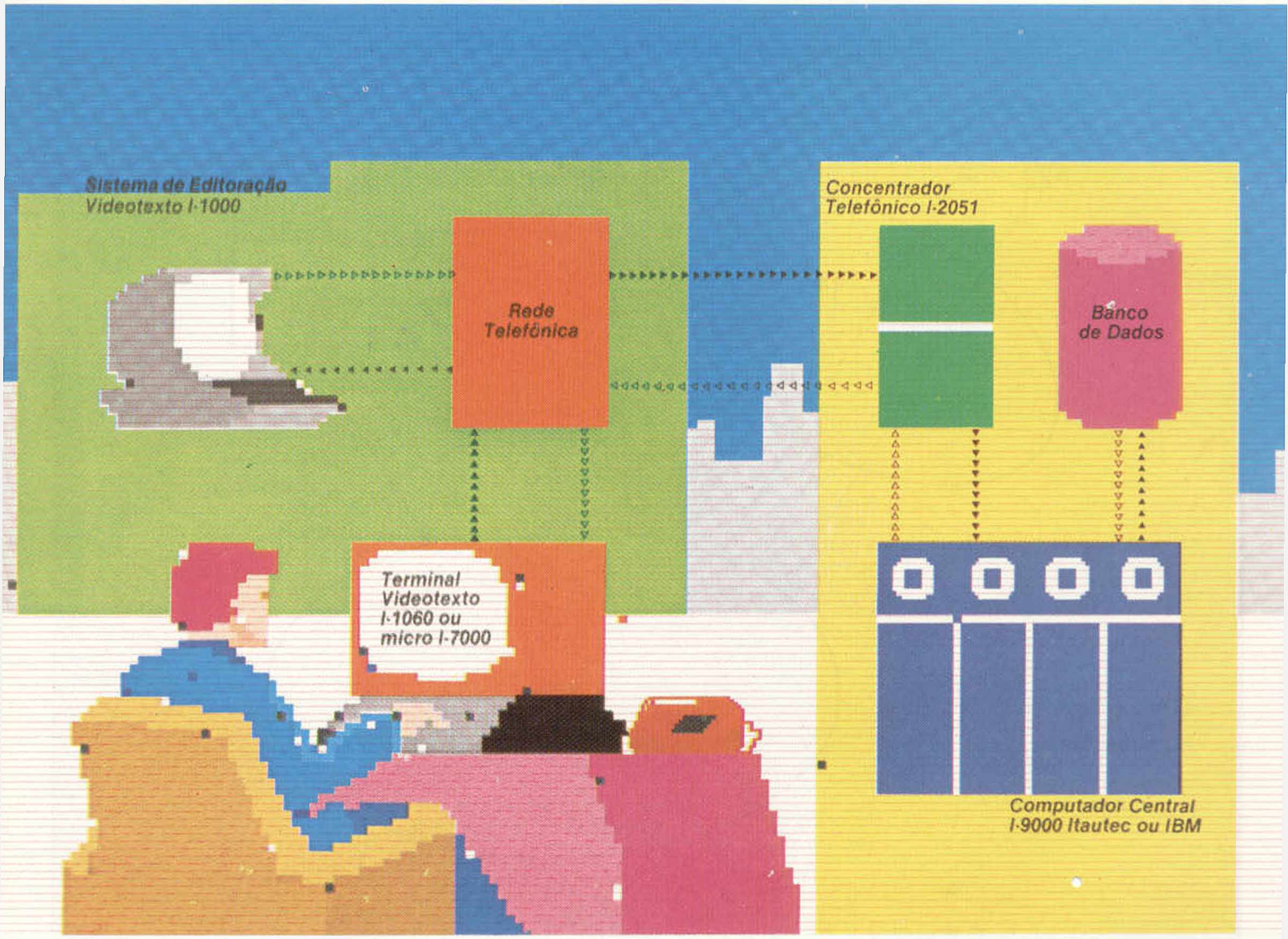
O fluxo de bytes que contém a informação desejada é convertido em impulsos elétricos, que percorrem a linha telefônica. Estes, atingindo o modem (conversor), a uma velocidade de cerca de 1.200 bauds (bytes por segundo), são reconvertidos em bytes. O modem reenvia então o fluxo de bytes ao terminal de videotexto, que o transforma em palavras e números exibidos no vídeo. Selecionam-se as informações desejadas pela digitação de palavras-chaves no teclado. Se possui uma impressora, o usuário pode obter uma cópia da informação recebida, ou então gravá-la em disquete ou fita cassete.

O videotexto inaugurou um caminho extremamente rápido de acesso a informações, tornando supérfluos tanto o acúmulo de papel quanto

Configuração definitiva para acesso ao Cirandão



A malha final do serviço Cirandão constituirá a Rede Nacional de Comutação de Pacotes - Renpac, e será complementada pelas redes públicas de telefonia e telex. T/M, terminal/micromputador; M, modem; CONC, concentrador.



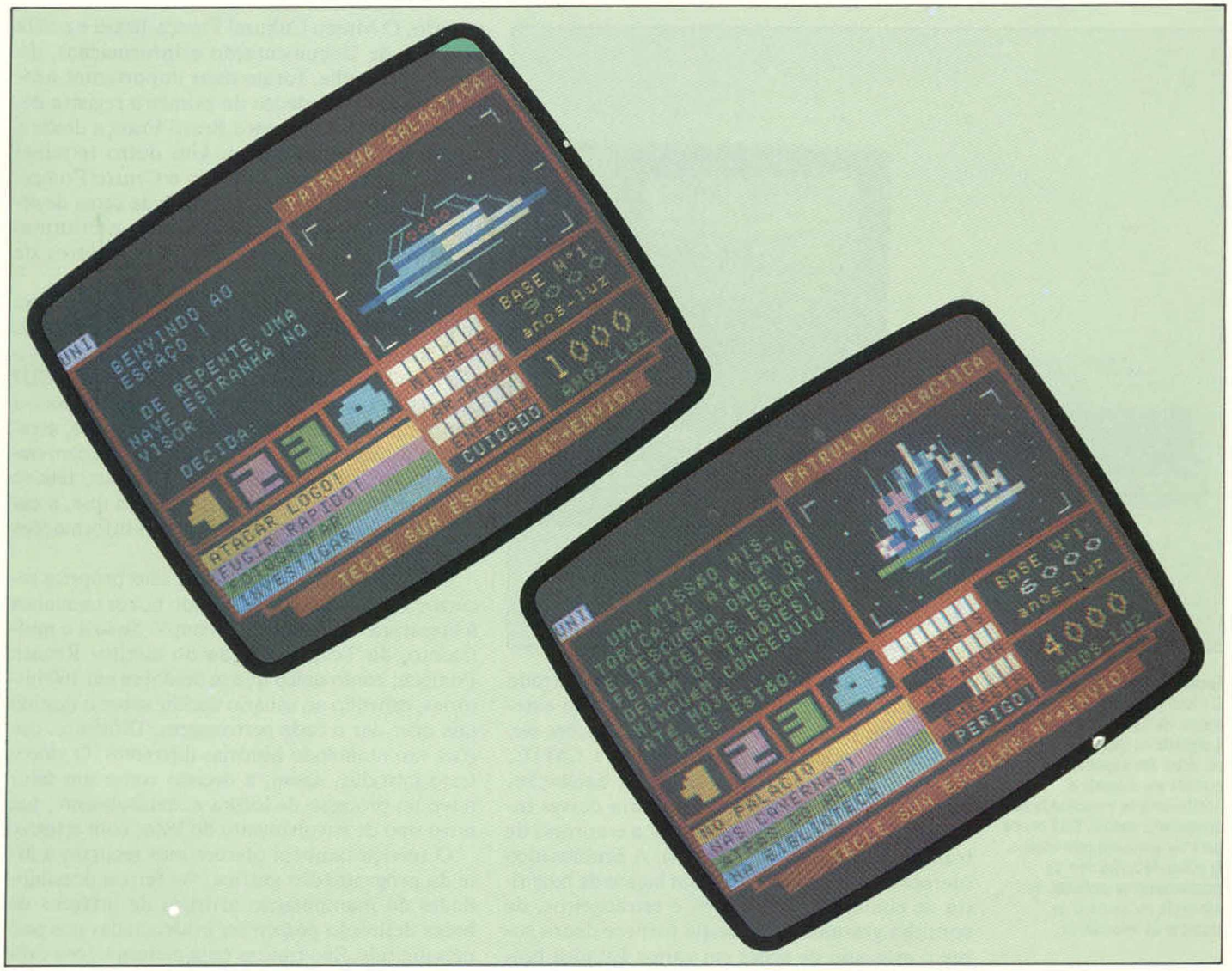
O sistema em uso
Esquema ilustrando como funciona o equipamento oferecido pela Itautec para serviços de edição de textos.

a lentidão associada às vias tradicionais de acesso à documentação escrita. Além disso, reduz bastante os prazos de várias formas de transação envolvendo papel, bem como os custos decorrentes dos métodos tradicionais para produzir impressos. Além disso, novas informações recebidas por um banco de dados são acrescentadas às já existentes ou substituem as mais antigas, mantendo o acervo sempre atualizado.

A quantidade de informações a que já se pode ter acesso, no Brasil, através dos atuais bancos de dados, chega a ser assombrosa. Não é apenas a extensão dessa rede que impressiona, mas também sua profundidade: em geral, acessam-se os registros, por meio de uma estrutura em árvore; ou seja, a cada opção feita, surgem novas opções, cada vez mais específicas. Diz-se que o usuário "navega" na árvore, seguindo o aprofundamento progressivo da informação. Navega-se por intermédio de comandos que manipulam menus e selecionam itens, números de páginas, palavras-chaves.

Espera-se que o progresso do videotexto seja acompanhado por uma correspondente evolução das formas de busca. Assim, a navegação pela árvore alia-se a elementos visuais no videotexto de segunda geração, cujos protótipos estão sendo estudados. Nele, o acesso às opções desejadas é feito através da imagem, assimilada a "caixas" que vão sendo sucessivamente abertas. Esse processo elimina a mediação do teclado. As caixas se abrem por meio de pontos na figura, ativados por periféricos como o joystick, o mouse ou a caneta óptica.

A Itautec, líder no mercado nacional de videotexto, oferece um equipamento completo de hardware e de software para gerenciar o sistema de editoração, abrigado em disquetes de 5 1/4 polegadas. O terminal para videotexto Itautec VDTX I-1060 compõe-se de adaptador, teclado, monitor ou televisor (opcionais) e cabos de conexão a gravadores. O primeiro manipula toda a eletrônica necessária para o uso do terminal, através de um módulo de controle, um microprocessador Z80, uma saída VHF (que permite ligar o terminal a um aparelho de tevê), uma saída RGB para vídeo, duas para gravador cassete e uma para fonte de alimentação.



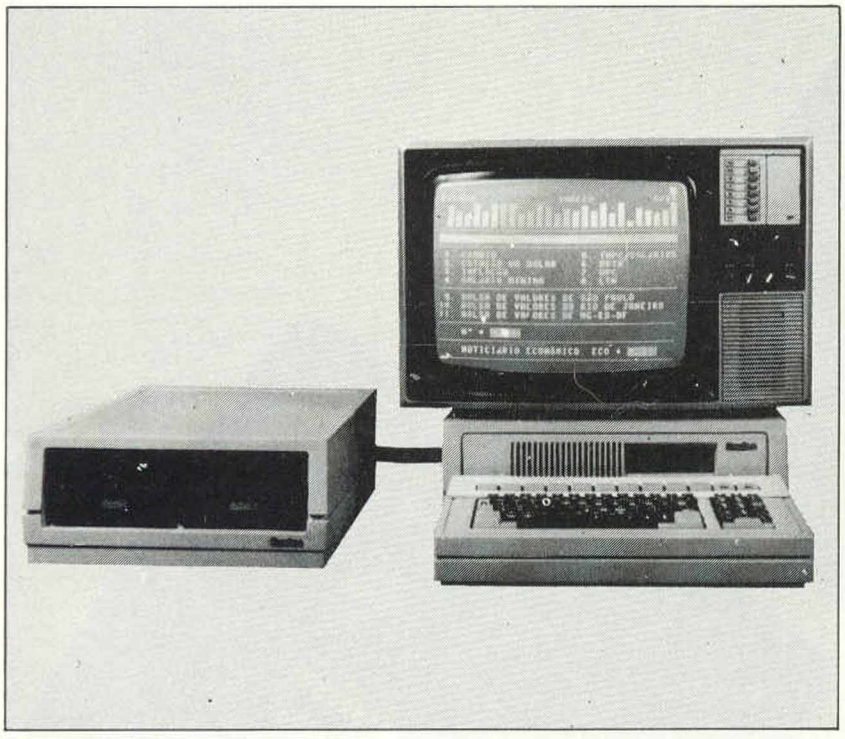
O teclado, padrão QWERTY, possui 44 teclas alfanuméricas, quinze de função e duas para futuras expansões. Além do terminal, a Itautec possui o sistema de editoração de videotexto I-1000, com o qual o fornecedor de serviços pode estruturar os programas de videotexto, criando, armazenando e atualizando as páginas exibidas no vídeo. A placa de interface para I-7072 contém todo o hardware necessário para transformar o micro I-7000 da Itautec num terminal para videotexto. É também parte integrante do sistema de editoração. Dois outros equipamentos, o concentrador telefônico I-2051 e a unidade de controle de comunicação UCCI I-4010, permitem a comunicação com terminais remotos.

No Brasil, alguns dos bancos de dados mais importantes são o do sistema Cirandão, da Embratel (que também oferece o sistema Interdata), o Aruanda, da Serpro, e o Videotexto, da Telesp. O procedimento de acesso ao Cirandão ilustra bem a facilidade de manipulação do equipamento. Discando um número apropriado de telefone, o usuário ouve um sinal contínuo. Transfere, então, a ligação para o modem. Após 5 segundos, digitará BREAK (ou NULL) para o sistema central, o qual, feita a conexão, pedirá ao usuário para se identificar através de uma senha. Pressionada a tecla [Enter], aparecerá o menu principal. Digitando [C] (Comunicações) nesse menu e [M] (de Mensagens Pessoais) no segundo, tem-se acesso à caixa postal, que permite receber ou enviar mensagens a outros usuários do Cirandão.

A importância dos serviços prestados pelo videotexto fica patente quando se pensa num banco de dados como o Cirandão Agropecuário, que fornece informações importantes sobre cultura e criação, instruções ao combate de pragas, preços de produtos e equipamentos, previsão do tempo etc., registros enviados por vinte instituições, além de vários centros de pesquisa. Outro importante banco de dados do sistema é o Cirandão Saúde, bem equipado em informações de várias organizações médicas e farmacológicas.

Já não há, no Brasil, praticamente nenhuma área de interesse público — ou mesmo restrito a setores mais especializados — desprovida da possibilidade de acesso a banco de dados. Assim, a Niterói Comércio Exterior e Despachos

Novidades à mão
O assinante do serviço de videotexto tem a seu alcance grande variedade de jogos e brincadeiras. O sistema permite acessá-los em segundos, carregá-los na memória do micro ou gravá-los.



Facilidade de impressão
O I-1000 para editoração de textos, da Itautec. Cada página é digitada no teclado e editada no vídeo. Em seguida, é gravada em disquete e posteriormente armazenada no computador central. Está pronta para ser acessada pelo usuário. O editor de textos vem se mostrando uma profissão em ascensão no mercado de trabalho da informática.

Aduaneiros Ltda. informa as datas de entrada e saída de navios, legislação do comércio exterior, custos sobre exportações e importações etc. A Prodam reúne informações sobre a CMTC, a Prefeitura de São Paulo, trânsito, habitação, multas e imposto predial (a maioria dessas informações, no entanto, é restrita a empresas de trânsito e secretarias de Estado). A Embravídeo oferece ao público em geral um banco de memória de comerciais brasileiros e estrangeiros, de consulta gratuita. A Bovespa fornece dados sobre o mercado de ações em vários Estados brasileiros. O banco de programas do Cirandão tem interesse para várias áreas administrativas, além de jogos e informações científicas e educacionais.

O serviço Interdata, da Embratel, permite o acesso a bancos de dados localizados no exterior. Um desses é o Dialog, que reúne informações de mais de 150 bases, com informações científicas, tecnológicas e econômicas, entre outras, como a história da América do Norte, educação, patentes e direitos autorais, e química. A Questel Telesystèmes, banco de dados francês, reúne 35 bases, como a Cancern, que informa sobre assuntos como bioquímica e imunologia, e a Cécile, onde se podem obter registros sobre desenho industrial, comunicação visual, arquitetura. A Dow Jones, que, como a Questel, pode ser acessada via Embratel, fornece resumos das notícias financeiras do *Wall Street Journal*.

A Telesp oferece vários serviços de interesse mais geral: roteiros turísticos, noticiário nacional e internacional, reservas de passagens em vôos nacionais, extratos de contas correntes do Bradesco e o serviço Teleshopping, que possibilita a compra de eletrodomésticos, brinquedos, artigos de cama e mesa etc. A utilização dos serviços da Telesp é cobrada na conta telefônica do usuário. O Museu Cultural França-Brasil e o SID (Sistema de Documentação e Informação), do Instituto Goethe, foram duas importantes adesões. O banco de dados do primeiro registra detalhes do relacionamento Brasil-França desde a época do Descobrimento. Um outro terminal comunica-se, via satélite, com o Centro Pompidou, em Paris, ligado a uma rede de cerca de setenta bancos de dados, com acesso a informações fornecidas por universidades, centros de pesquisa e bibliotecas.

Aliás, a França, um dos países pioneiros na implantação de videotexto, já conta com mais de 2 milhões de terminais instalados, e a previsão é de 11 milhões até o fim da década. O SID do Instituto Goethe comunica-se on line com o banco de dados central da Alemanha; este, através do sistema INKA, troca informações com outros bancos do mundo inteiro. Assim, tem-se acesso a 15 milhões de itens, volume que, a cada ano, aumenta em 500.000 novas informações científicas e culturais.

O videotexto pode emprestar seus próprios recursos à palavra escrita, abrindo novos caminhos à literatura. Interessante exemplo disso é o multiconto, da Telesp, criação do escritor Renato Pompeu; conto único que se desdobra em 160 histórias, cabendo ao usuário decidir sobre o destino que quer dar a cada personagem. Diferentes opções vão montando histórias diferentes. O videotexto introduz, assim, a decisão como um fator novo no processo de leitura e, naturalmente, um novo tipo de envolvimento do leitor com o texto.

O serviço também oferece seus recursos à arte da programação gráfica. As férteis possibilidades da manipulação artística de imagens de baixa definição podem ser evidenciadas nos painéis digitais. São figuras cuja metamorfose e diluição têm tanto peso quanto elas próprias. Artistas como Júlio Plaza e poetas como Décio Pignatari são pioneiros em explorar a arte desse "videotexto de imagens". É provável que um desenvolvimento, nesse novo meio de comunicação, de horários "reservados a comerciais", promova muito essa área, ainda tímida do videotexto; tanto como poderá também promover e afetar os próprios rumos do videotexto. A importância desse novo meio na área educativa é outro motivo que certamente fará da arte da programação gráfica um poderoso auxiliar.

A esses fatos alia-se a ênfase que o videotexto de segunda geração pretende dar à programação gráfica. As perspectivas para o futuro são ainda mais fascinantes. A alta resolução da imagem por pixels revela-se um campo no qual a pesquisa caminha a passos largos. Imagens digitais de alta definição podem ser ampliadas localmente — aliás, uma das características mais curiosas da televisão digital. Hoje, o custo para tal implantação seria impraticável, mas é possível que o videotexto do futuro comporte bancos de dados fotográficos. O usuário teria então acesso ao acervo dos museus de arte usufruindo as pinturas por meio de ampliações locais.

RASCUNHO ELETRÔNICO

A pilha é uma área de trabalho que permite, através de instruções próprias, copiar e restaurar o conteúdo dos registradores. Examinamos aqui a pilha e suas operações, essenciais na execução de sub-rotinas em ASSEMBLY.

A essência da linguagem ASSEMBLY é a manipulação da memória; daí porque as instruções até aqui estudadas servirem, na maioria, basicamente para carregar dados em determinadas posições de memória, ou transferi-los para outros locais. A maneira de acessar tais posições varia conforme o modo de endereçamento, mas o endereço sempre faz parte do operando. Existem, entretanto, certas instruções que acessam uma área da memória, mas não usam o endereço como operando. Elas operam na área de memória conhecida como pilha, motivo pelo qual se chamam operações de pilha.

A pilha — uma área temporária de trabalho, disponível tanto para o programador como para a CPU — é um tipo de bloco de rascunho, onde facilmente se lêem, gravam e apagam informações. As operações de pilha copiam dados dos registradores nas áreas desocupadas da pilha, ou desta naqueles. Tais instruções não requerem endereços como operandos, pois há um registrador específico para esse fim — o ponteiro da pilha —, que sempre contém o endereço da próxima posição livre na pilha. Assim, qualquer comando para gravar dados na pilha irá utilizar o byte indicado pelo ponteiro, e os comandos que copiam dados da pilha irão acessá-los no endereço onde foram gravados dados pela última vez. A execução de qualquer operação de pilha já inclui o reajuste do ponteiro.

Nos equipamentos baseados no processador 6502, a pilha ocupa 256 bytes de RAM, da posição $0100 até $01FF; no Z80, a localização e o tamanho da pilha são determinados pelo sistema operacional, e modificáveis pelo programador. Essa variação reflete as diferenças na organização interna dos dois microprocessadores: o ponteiro de pilha do 6502 tem um único byte, enquanto o do Z80 tem 2 bytes.

No 6502, a CPU considera o conteúdo do ponteiro como o byte inferior do endereço da pilha; a ele se acrescenta automaticamente um byte superior de $01, por meio de um "nono bit" associado ao ponteiro. Esse bit extra sempre tem o valor 1 e, assim, os endereços de pilha no 6502 ficam sempre na página 1.

O ponteiro de pilha do Z80 é um registrador de 2 bytes, capaz de indicar qualquer localização entre $0000 e $FFFF — ou seja, todo o espaço endereçável do próprio processador Z80. Assim, a pilha pode situar-se em qualquer lugar da RAM, sendo sua posição alterável pelo programador. Mas não se recomenda tal procedimento, pois o sistema operacional inicializa a localização da pilha e nela armazena dados. O sistema operacional pode interromper a qualquer momento a execução de um programa em código de máquina e, nesse caso, irá procurar na pilha os dados necessários para sua operação; assim, qualquer mudança na localização da pilha implicará a ausência dos dados relevantes, resultando até num colapso do sistema.

Para exemplificar o uso da pilha, veja a rotina abaixo, que troca o conteúdo de dois locais de memória, LOC1 e LOC2. As instruções PUSH e POP do Z80 indicam, respectivamente, a ação de colocar e a de retirar dados da pilha.

6502		Z80	
LDA	LOC1	LD	A,(LOC1)
PHA		PUSH	AF
LDA	LOC2	LD	A,(LOC2)
STA	LOC1	LD	(LOC1),A
PLA		POP	AF
STA	LOC2	LD	(LOC2), A

Inicialmente se carrega o conteúdo de LOC1 no acumulador; deste, eles são copiados para a pilha. Em seguida, carrega-se no acumulador o conteúdo de LOC2, que daí é armazenado em LOC1. Copia-se então o conteúdo do byte superior da pilha para o acumulador, que passa assim a armazenar o conteúdo original de LOC1. Transfere-se então este para LOC2, completando-se a troca. Note que as operações de pilha "salvaram" o conteúdo de LOC1 na memória pelo tempo necessário, mas o programa não especificou nenhuma posição para esse fim. Por implicação, essa posição foi suposta como a próxima localização livre da pilha.

Nosso programa-exemplo esclarece vários outros pontos sobre as operações de pilha. Basicamente, elas são recíprocas e seqüenciais. O último item colocado na pilha é recuperado pela próxima operação de extração. Se houver várias introduções de dados sem nenhuma extração, os dados irão sendo gravados em posições sucessivas da pilha, um "em cima" do outro; da mesma forma, havendo várias extrações de dados sem qualquer acréscimo, os dados irão sendo acessados "de cima para baixo" na pilha.

Para visualizar melhor esse processo, imagine-se escrevendo vários cartões postais e empilhando-os à sua frente; ao terminar, irá ler um por um até acabar a pilha, antes de enviá-los. O último cartão escrito estará no topo da pilha. Esse tipo de estruturação dos dados chama-se LIFO (Last In First Out, "último a entrar, primeiro a sair"). O inverso — a FIFO (First In First Out, "primeiro a entrar, primeiro

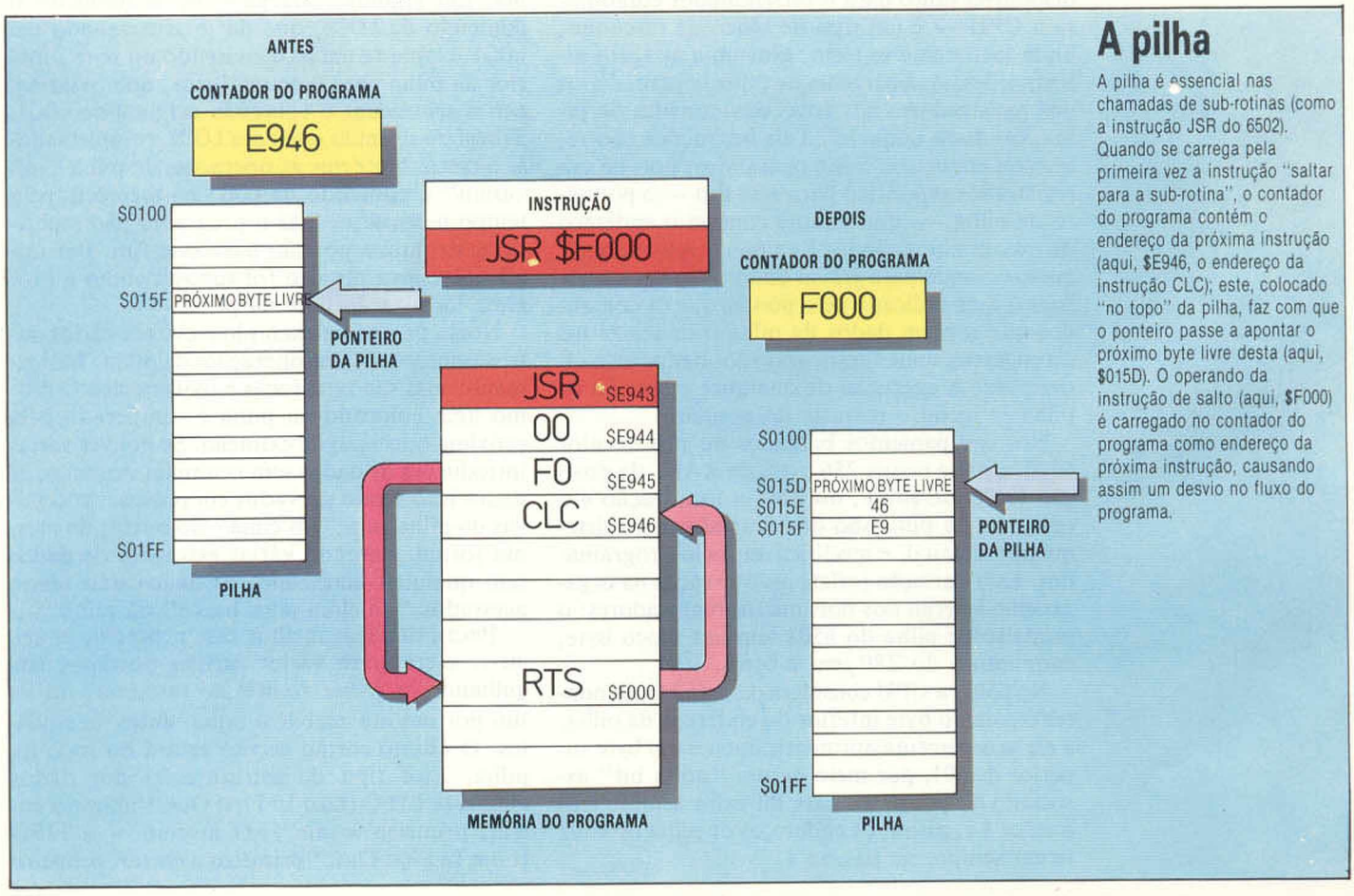
a sair") — é uma simples fila, como as que enfrentamos no dia-a-dia. Apenas por convenção considera-se que a pilha cresce para cima e que o próximo byte livre seja o do topo. Na verdade, tanto no Z80 como no 6502, cada introdução de dados irá decrementar o ponteiro, de forma que o topo da pilha tem um endereço de memória inferior ao da base. Podemos dizer que a pilha "cresce em direção a zero".

Também observamos na rotina de exemplo que o número de instruções que introduzem dados é idêntico ao das que extraem dados. Normalmente se trabalha com pilhas dessa maneira; não é essencial, mas, se desrespeitarmos esse equilíbrio entre os opostos, as sub-rotinas que escrevemos poderão ter um retorno incorreto, ocasionando falha no programa. Esse é um dos erros mais comuns nos programas em ASSEMBLY, mas de fácil correção: basta comparar no programa o número de instruções de introdução com o de extração de dados.

Há grande diferença entre a versão Z80 e a 6502: esta sempre coloca na pilha registradores de um só byte; aquela, de 2 bytes. No Z80, ao se introduzir ou extrair o conteúdo do acumulador, também se introduz ou se extrai o conteúdo do registrador de estado de processador. Isso porque a CPU trata esses dois registradores de um só byte como um só registrador de 2 bytes, chamado registrador AF (Accumulator Flag, "flag do acumulador"). A potência do Z80 deriva em grande parte de sua capacidade de manipular registradores de 2 bytes.

A boa técnica de programação recomenda, ao se iniciar uma sub-rotina, que se coloque na pilha o conteúdo de todos os registradores da CPU e, logo antes de se retornar dela, que se retire esse conteúdo. Isso vai assegurar que a CPU, após a execução da sub-rotina, continue num estado idêntico ao anterior. Significa também que se pode utilizar na sub-rotina qualquer dos registradores sem risco de adulterar dados essenciais ao programa. Por exemplo, considere esta sub-rotina de um programa:

	6502		Z80	
	LDA	LOC1	LD	A,LOC1
SUM	ADC	#$6C	ADC	A,$6C
GSUB	JSR	SUBR0	CALL	SUBR0
TEST	BNE	SUM	JR	NZ,SUM
EXIT	RTS		RET	
SUBR0	PHP		PUSH	AF
	PHA		PUSH	HL
	TXA		PUSH	DE
	PHA		PUSH	BC
	TYA		PUSH	IX
SUBR1	PHA		PUSH	IY
SUBR2	STA	LOC2	LD	(LOC2),A
	LDA	#$00	LD	A,$00
SUBR3	PLA		POP	IY
	TAY		POP	IX
	PLA		POP	DE
	TAX		POP	BC
	PLA		POP	HL
SUBR4	PLP		POP	AF
	RTS		RET	



A pilha

A pilha é essencial nas chamadas de sub-rotinas (como a instrução JSR do 6502). Quando se carrega pela primeira vez a instrução "saltar para a sub-rotina", o contador do programa contém o endereço da próxima instrução (aqui, $E946, o endereço da instrução CLC); este, colocado "no topo" da pilha, faz com que o ponteiro passe a apontar o próximo byte livre desta (aqui, $015D). O operando da instrução de salto (aqui, $F000) é carregado no contador do programa como endereço da próxima instrução, causando assim um desvio no fluxo do programa.

As instruções entre SUBR0 e SUBR1 colocam na pilha o conteúdo atual do registrador; as instruções entre SUBR3 e SUBR4 restauram nos registradores esse conteúdo. As instruções essenciais são as duas que iniciam em SUBR2, mas a segunda delas não tem efeito, uma vez que as subseqüentes alteram por completo o estado do acumulador.

Note que as instruções PUSH e POP do Z80 tomam como operando qualquer dos pares de registradores; o 6502, contudo, trabalha apenas com o acumulador (PHA e PLA) e com o registrador de estado do processador (PHP e PLP). Daí vem a necessidade, na versão 6502, de transferências entre registrador e acumulador (TXA, TAX, TYA, TAY). Note também que, na versão Z80, cometemos deliberadamente um erro ao não extrair todos os registradores na ordem inversa em que foram colocados na pilha. Isso ilustra o cuidado que se deve tomar em operações de pilha, e demonstra que é possível introduzir dados a partir de um registrador e extraí-los de volta para outro — uma maneira trabalhosa, mas às vezes conveniente, de transferir dados entre registradores.

No próximo artigo examinaremos os usos e funções dos registradores da CPU, e iniciaremos o estudo da aritmética do código de máquina.

Exercícios

A) Reescreva a segunda rotina dada nas "Respostas aos exercícios anteriores", de modo que a mensagem em LABL1 seja armazenada de novo no mesmo local, mas na ordem inversa, assim:

LABL1 MEGASNEM AMU SIE

Use a pilha para essa inversão.

B) Desenvolva essa rotina para que as palavras da mensagem permaneçam em sua ordem original, mas com os caracteres de cada palavra invertidos, assim:

LABL1 SIE AMU MEGASNEM

Respostas dos exercícios anteriores

A) Esta sub-rotina armazena os números de $0F a $00, em ordem decrescente, no bloco de $10 bytes reservado pelo pseudocódigo DS em LABL1.

6502		Z80	
ORIGIN	ORG $7000	ORIGIN	ORG $C000
LABL1	DS $10	LABL1	DS $10
LABL2	DW $7100	LABL2	DW $C100
		OFFST	EQU $0F
BEGIN	LDY #$FF	BEGIN	LD IX,LABL1
	LDX #$10		LD B,OFFST
LOOP0	INY	LOOP0	LD (IX+0),B
	DEX		INC IX
	TXA	ENDLP0	DJNZ LOOP0
	STA LABL1,Y		LD (IX+0),B
ENDLP0	BNE LOOP0		RET
	RTS		

O 6502 usa o registrador Y como índice para o endereço LABL1 e o registrador X como contador do loop e fonte dos dados a serem armazenados. Note que o registrador X é decrementado duas instruções antes do teste BNE em ENDLP0. Entretanto, como STA e TXA (tranferir o conteúdo de X para o acumulador) não afetam o registrador de estado do processador, o teste verifica o efeito da redução de X.

A versão Z80 usa o modo de endereçamento indireto IX para manter o endereço de armazenamento e o registrador B como contador e fonte de dados. Em ENDLP0, vemos DJNZ LOOP0, que significa "decrementar o registrador B e dar um salto relativo para LOOP0 se o resultado for diferente de zero". Essa instrução quase corresponde, em ASSEMBLY, à estrutura FOR-NEXT, e certamente torna mais fácil e conveniente escrever loops no Z80.

B) Essa rotina copia a mensagem armazenada em LABL1 para o bloco que inicia no endereço armazenado em LABL2. Armazena-se no fim da mensagem, como indicador de término, o valor $0D (o código ASCII para Return ou Enter).

6502		Z80	
ORIGIN	ORG $7000	ORIGIN	ORG $C000
LABL1	DB 'EIS UMA MENSAGEM'	LABL1	DB 'EIS UMA MENSAGEM'
TERMN8	DB $0D	TERMN8	DB $0D
LABL2	DW $7100	LABL2	DW $C100
CR	EQU $0D	CR	EQU $0D
ZPL0	EQU $FB		
BEGIN	LDA LABL2	BEGIN	LD IX,LABL1
	STA ZPL0		LD IY,(LABL2)
	LDA LABL2+1	LOOP0	LD A,(IX+0)
	STA ZPL0+1		LD (IY+0),A
	LDY $FF		INC IX
LOOP0	INY		INC IY
	LDA LABL1,Y		CP CR
	STA (ZPL0),Y	ENDLP0	JR NZ,LOOP0
	CMP CR		RET
ENDLP0	BNE LOOP0		
	RTS		

A versão 6502 usa o registrador Y como índice para o endereço indireto ZPL0 no modo de endereçamento pós-indexado. Esse modo é possível apenas com o registrador Y e requer um endereço de operando de página 0 — daí a inicialização de ZPL0 e ZPL + 1 com o endereço armazenado em LABL2. O sistema operacional das máquinas baseadas no 6502 usa a maioria das posições da página 0. A versão Z80 usa IX no modo indexado e IY no modo indexado indireto.

Ambas as rotinas usam uma instrução "comparar o acumulador" — CMP CR (6502) e CP CR (Z80) — na qual se subtrai o operando do conteúdo do acumulador, afetando assim os flags do registrador de estado do processador (PSR). Restaura-se então o conteúdo do acumulador, enquanto o PSR mostra os resultados da comparação. Quando o acumulador contém $0D (o indicador de término de mensagem), o resultado da comparação é a ativação do flag 0. Assim, o teste ENDLP0 falhará, passando o controle à instrução de retorno.

MICROSOFT

A Microsoft tornou-se um dos maiores fornecedores de software para microcomputadores, e participou da formulação das especificações do IBM PC, o mais vendido do mundo.

A Microsoft, hoje uma empresa milionária, é uma história clássica de entusiasmo bem-sucedido. Bill Gates, de 28 anos, seu presidente, em 1972 era apenas amador talentoso.

Na Seattle High School — que a associação de pais e mestres teve a feliz idéia de equipar com um terminal de computador ligado a um míni tipo DEC PDP-11 —, Bill aprendeu as noções básicas de informática. Estudou depois na Universidade de Harvard, onde obteve sua graduação. Ao voltar a Bellevue, iniciou-se nos negócios com o colega de universidade Paul Allen.

A empresa que então fundaram chamava-se Traff-D-Data, cujo trabalho consistia em monitorar o fluxo do tráfego para o departamento de trânsito de Seattle. Ocorria um período de desenvolvimento extraordinário do microcomputador: os primeiros microprocessadores estavam surgindo e quem possuía um pouco de visão e entusiasmo percebeu o grande futuro dos chips, como o Intel 4004 e mais tarde o 8008. Na ocasião, Bill já estava inteiramente familiarizado com o míni DEC PDP-11, e um de seus primeiros objetivos foi tentar localizar deficiências nesse equipamento. Ocorreu-lhe que seria uma boa idéia adaptar o BASIC desse míni para uso com o chip. Ele não tinha um sistema de desenvolvimento, e a primeira ocasião em que o BASIC rodou na máquina foi quando Gates levou as fitas para Altair, em Albuquerque, Novo México. Incrivelmente, rodou sem problemas logo da primeira vez. Assim nasceu a linguagem MBASIC, que passou a ser empregada como padrão, até agora insuperado.

A Microsoft estava ficando conhecida como uma software house especializada na adaptação de sistemas operacionais a novos computadores — preencher buracos, por assim dizer. Foi quando a IBM entrou em contato com Gates, solicitando seu conselho sobre como configurar e equipar um computador pessoal.

Inicialmente Gates sugeriu que Gary Kildall, da Digital Research — então no auge da fama, com o florescente sucesso do sistema operacional CP/M —, seria o homem indicado para o serviço. Mas a IBM voltou a procurá-lo, e a Microsoft acabou reescrevendo as linguagens PASCAL, FORTRAN e MBASIC para micros de 16 bits, além de criar a linguagem GU (de "geewhizz", uma exclamação de surpresa) BASIC, com capacidades musicais e gráficas ampliadas.

Ao mesmo tempo, Gates percebeu que um sistema operacional multiusuário — um tanto desorganizado, mas potente — da Bell Laboratories poderia ser adaptado para os micros mais poderosos baseados nos novos microprocessadores de 16/32 bits. E transformou o Unix no Xenix, que a Tandy e a Apple adotaram em seus próprios modelos de 16/32 bits, em 1983. Chegou a transpirar que a Microsoft teria sido responsável por grande parte da nova criação da Apple, o microcomputador Macintosh.

A Microsoft tem participação apreciável também no mercado amador. Em 1981, ela fundou a ASCII-Microsoft e colocou um talentoso jovem japonês, Kay Nishu, para vender seus sistemas operacionais e o BASIC aos fabricantes de micros portáteis do Extremo Oriente — por exemplo, o NEC PC 8201 e o Tandy Model 100. Como resultado do desejo dos fabricantes japoneses de características comuns, não somente para as linguagens, mas também nas interfaces para periféricos, desenvolveu-se o padrão MSX. O próximo passo da Microsoft, ao que tudo indica, será o formato padronizado para discos, que possibilitará a transferência de informações entre os três principais ambientes operacionais — o MSX, o MS-DOS e o Xenix.

Com sua ênfase em softwares fáceis de usar, e baseados em recursos avançados como as janelas na tela e o mouse, a Microsoft parece ter pela frente um brilhante futuro.

PRINCÍPIOS CONDUTORES

Em 1970, com a idade de 28 anos, Shiina Takayoshi abandonou uma promissora carreira militar e fundou a Sord Corporation. Ele imediatamente formulou onze princípios para dirigir sua nova empresa de computação. Alguns desses princípios eram os seguintes:

- **A principal obrigação da companhia é para com a humanidade.**
- **A companhia deve esforçar-se ao máximo a fim de determinar quais os melhores produtos e serviços para a sociedade, e fornecê-los a um custo razoável.**
- **Não haverá qualquer divisão entre os funcionários e a administração. Todas as pessoas da companhia devem se respeitar mutuamente e cooperar para o benefício de todos.**

Padrão da indústria

A linguagem BASIC — (Beginners' All-purpose Symbolic Instruction Code, "código de instrução simbólica para todos os fins, para principiantes") — foi desenvolvida em 1965, no Dartmouth College, EUA, por J. Kemeny e T. Kurtz; antecede, pois, o microprocessador em pelo menos sete anos. Embora muitas versões dessa linguagem tenham sido criadas, o MBASIC (a versão da Microsoft) acabou reconhecido como o padrão industrial.

A Microsoft firmou sua reputação no mercado em cima do sucesso do MBASIC e continuou a prosperar, produzindo um sério desafiante ao CP/M — o MS-DOS, um sistema operacional projetado para aplicação numa ampla gama de micros.

Seguindo a liderança determinada pelo sistema terminal Star, da Xerox — desenvolvido pela Apple com o Lisa e o Macintosh —, a Microsoft produziu um pacote combinando o software que divide a tela em janelas independentes (o MS-WINDOWS) e o dispositivo para manejá-las: um mouse com uma esfera conjugada a dois seletores para deslocar o cursor pela tela.

perseguir o herói, reduzindo sua liberdade de ação.

Para o programador, os jogos de plataforma apresentam várias vantagens. A primeira delas é a necessidade de relativamente pouca codificação. Definido o desenho gráfico do herói, dos guardiões, plataforma, escadas e elevadores, todas as diferentes "salas" poderão ser programadas apenas por reposicionamento dos vários elementos. Isso poupa muito espaço de memória, que pode então ser usado para realçar detalhes do jogo ou aumentar o número de salas. É possível economizar ainda mais memória, pois cada plataforma não requer armazenamento em separado. O programador só precisa gravar o desenho gráfico e as listagens, armazenando separadamente apenas o comprimento de cada plataforma.

Impossible Mission tira vantagem dessa codificação característica de um jogo de plataforma. Seu objetivo consiste em encontrar as várias partes de um código — escondidas nos móveis de 32 salas — que permite ao jogador penetrar no reduto do perverso cientista Elvin.

Os móveis são guardados por robôs, ocasionalmente por uma grande bola negra. Dentro de alguns móveis há chaves especiais que, uma vez conseguidas, permitem acionar os elevadores nas salas ou desligar temporariamente os robôs, para passar por eles.

Essas chaves encontram-se em terminais de computador, presentes em cada uma das salas. O jogador passa de uma sala a outra por um elevador e por uma série de corredores. No canto esquerdo inferior da tela, um mapa mostra as salas revistadas até então e a posição atual do jogador.

Também se podem obter chaves em salas de código. A sala emite várias notas, de intensidade variável; e, para conseguir uma chave, o jogador tem de arranjá-las em ordem ascendente. Quanto maior o número de chaves que se tenta obter, maior o número de notas emitidas. Nesse sentido, Impossible Mission se mostra um jogo de plataforma padrão. Mas, ao contrário de muitos outros do mesmo tipo, aqui o jogador não tem um número determinado de "vidas"; em vez disso, sempre que o herói é "morto", o jogador sofre uma penalidade de tempo.

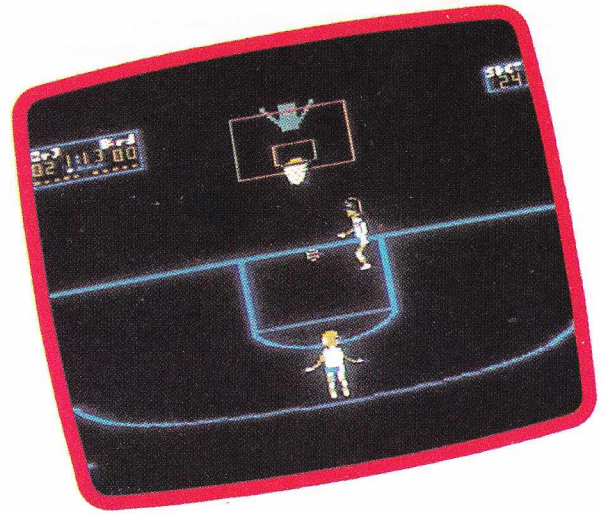
O espaço de memória poupado pelos programadores tem bom aproveitamento aqui. A característica mais importante nesse sentido é a síntese de fala, notável pelas grandes quantidades de bytes que ocupa — duas frases curtas, com cerca de dez palavras cada uma, ocupam até 3 ou 4 Kbytes de RAM. Nesse particular, Impossible Mission está entre os melhores e mais claros exemplos conhecidos.

A definição dos desenhos também é muito boa, e, embora as salas não sejam totalmente preenchidas por eles, os detalhes individuais do herói, dos móveis e dos robôs são minuciosos. Esses vários elementos, somados, resultam num jogo muito bem elaborado e divertido.

Impossible Mission: Para os computadores Commodore 64 e Atari.
Editor: CBS Software, Inglaterra.
Autor: Dennis Caswell.
Joystick: Necessário.
Formato: Disco ou cassete.

UM-CONTRA-UM

One-On-One (análise)



À medida que cresce a popularidade do basquetebol em todo o mundo, aumenta a oferta de softwares prontos para esse jogo, especialmente na modalidade um-contra-um, que permite ao usuário de micros enfrentar sozinho o computador.

Não surpreende que muitos pacotes de jogos comercializados nos Estados Unidos se baseiem nos esportes populares americanos. Mas muitas dessas versões para computadores também são apreciadas em outros países. É o caso do basquetebol, embora a modalidade envolvendo dois times não seja a única existente.

One-On-One, que opõe apenas um jogador contra outro e usa só meia quadra, consiste numa alternativa para o jogo, especialmente pela dificuldade em se mobilizarem dez participantes. Também é um ótimo treinamento para os fundamentos do basquete.

O One-On-One tenta apreender o espírito desse esporte, produzindo um pacote endossado por duas das figuras mais populares do basquete profissional americano: Julius Erving (o famoso "Dr. J") e Larry Bird ("Big Bird").

As ações e movimentos dos dois jogadores na tela, embora desenhados de forma não espetacular, são bastante precisos — e perfeitos no gancho que tornou "Big Bird" famoso.

Todas as regras do basquete foram incorporadas, igualando-se o estilo de cada jogador, embora isso só seja percebido pelos torcedores ou praticantes mais fanáticos.

Há quatro níveis de dificuldade no One-On-One, e várias maneiras diferentes de escolhê-los. Eles dependem das variações nas regras, de o jogo ser para um ou dois jogadores e de qual bonequinho é controlado pelos jogadores ou pelo computador.

O tempo de arremesso é de apenas 24 segundos para o atacante, que sempre deve prestar atenção na "barra de cansaço", e ter em mente as regras gerais do jogo, especialmente quanto a faltas.

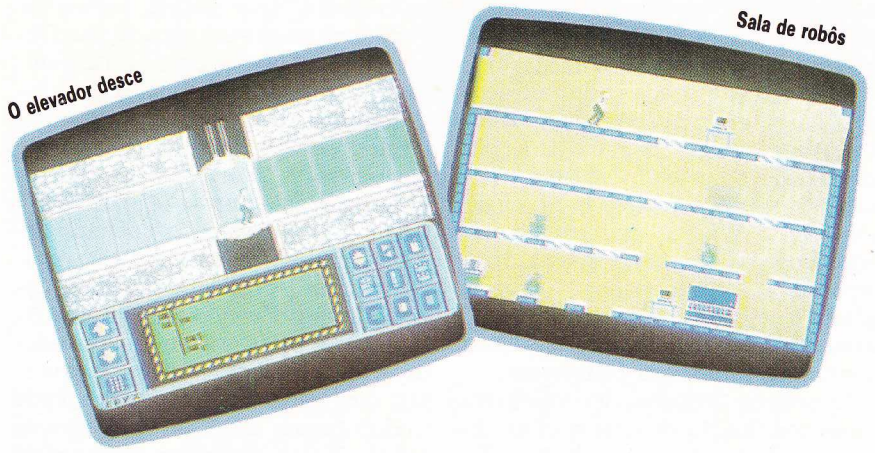
O controle da bola é quase perfeito, e o arremesso se faz simplesmente pressionando o botão de disparo. Entre outras coisas, podem ser executados movimentos tais como iniciar e completar um jump e fazer o corta-luz, girando 180 graus para tentar driblar o adversário.

Se o jogador estiver na defesa, poderá roubar a bola, tentar um rebote ou cortar um lançamento, se bem que, ao atuar contra o computador, o usuário será muito exigido na interceptação de arremessos. Entretanto, uma vez no ataque, terá à disposição uma grande variedade de movimentos, suficientes para que a máquina propicie uma boa satisfação pelo dinheiro dispendido na aquisição do pacote.

One-On-One: Para os computadores Commodore 64, Atari e Apple II.
Editor: Ariolasoft, Estados Unidos.
Formato: Disco.
Joystick: Necessário.

CAÇA AO CÓDIGO SECRETO

Impossible Mission (análise)



Nada é impossível
O objetivo de Impossible Mission é percorrer, por meio de elevadores e corredores, as várias salas do reduto de um perverso cientista. Enquanto evita os robôs, o jogador deve procurar as partes da chave que lhe permitirá entrar no laboratório.
Ocasionalmente o jogador também pode conseguir "sedadores", dispositivos que desligam os robôs por algum tempo.

Numa corrida contra o tempo, o jogador deve penetrar na fortaleza do diabólico Elvin, reunindo pistas para chegar a um código valioso. Impossible Mission coloca o participante numa posição nada invejável, mas muito divertida, com gráficos vistosos e síntese de fala.

Trata-se de um programa de jogo que faz parte do gênero plataforma, muito popular. O jogador (no papel de herói) deve obter itens posicionados em vários pisos nas inúmeras telas. O personagem normalmente entra pelas extremidades inferiores do vídeo e deve atingir as plataformas através de uma série de elevadores, escadas e trampolins (ou qualquer outra coisa na qual tenha pensado o programador).

Com freqüência, colocam-se elementos vitais, como certos achados ou pistas importantes, em locais particularmente incômodos, o que leva a raciocinar bastante sobre como chegar até eles.

Tais jogos são complicados ainda mais pela presença de alienígenas ou outras figuras gráficas perigosas, que devem ser evitadas, pois geralmente é fatal tocá-las.

Essas figuras podem ser estáticas ou movimentar-se dentro de padrões predefinidos. Além disso, muitos desses sprites têm capacidade de "fogo". Dependendo das intenções do programador, os movimentos dos inimigos mostram-se mais ou menos inteligentes. Muitos deles se movem para a frente e para trás, ao longo de uma só trajetória; já outros são programados para