Materializando abstrações: O uso de Algoritmos Vivos em sala de aula

Pedro Luis Kantek Garcia Navarro

kantek@pr.gov.br

CELEPAR / Universidade Positivo Al. Dr. Carlos de Carvalho 250, ap52 80410-180 Curitiba PR

O currículo LATTES do autor pode ser visualizado em http://lattes.cnpq.br/7330616928412664

Resumo

Comenta-se a dificuldade de ensinar algoritmos e programação. Dificuldade esta, crescente pela universalização de acesso aos cursos universitários de informática. Aumentando em número, tais cursos agregam mais e mais interessados, porém o desempenho médio desses alunos parece diminuir. Vem também a constatação de que o uso do computador na empreitada de ensinar programação é pobre: limita-se a ser compilador e executor de programas. Apresenta-se a idéia de algoritmos vivos, como ferramenta para materializar um algoritmo: o abstrato torna-se concreto. Princípios de seu uso e funcionamento são a seguir apresentados. Um exemplo de algoritmo vivo usado para transmitir as idéias básicas da computação evolutiva é apresentado. A lista dos mais de 200 algoritmos vivos existentes é mostrada e comentada. Conclui-se pela pertinência de seu uso e melhora dos resultados obtidos em situações reais de aprendizado de programação de computadores.

Palavras chave

programação, algoritmos, prática de programação, pedagogia da informática, ensino de algoritmos, ensino de programação.

Antecedentes

Poucas tarefas são tão difíceis como ensinar a projetar, escrever, implantar, depurar e manter algoritmos. Seja porque são meras abstrações, ou porque trata-se de matemática aplicada ou ainda porque ao adquirir "vida" os algoritmos tornam-se mais e mais complexos de serem apreendidos, o caso é que apenas uma pequena parcela dos alunos é que pode ser apresentada aos algoritmos e ato contínuo, sair os criando e os usando. Tom de Marco descreve em seu livro Peopleware, que é normal encontrar diferenças na produtividade de programadores de 10:1 [1]. Trazendo esta diferença de desempenho para dentro da sala de aula, percebe-se que há que se usar de criatividade para permitir algum aprendizado a todas as pessoas, independente de sua localização no espectro de capacidade e produtividade acima citado.

Para os que formam a maioria e que se localizam nas faixas menores de produtividade, o aprendizado é dificil e demorado. Muitos exercícios há que fazer e muitas horas passar diante de um computador para que este, pacienciosamente – embora com teimosia – vá mostrando como são as entranhas de um algoritmo, o que o move e como os resultados (in) esperados são produzidos.

Há que estimular os alunos, instigá-los e indicar os caminhos do aprendizado. A simples colocação de conceitos, algoritmos, casos exemplo e princípios de funcionamento na expectativa de que esses conceitos e princípios sejam absorvidos pelos alunos, tem se mostrado expectativa quase irrealizável. Muitos dos alunos parecem não saber "por onde começar".

Este estado de coisas tem sido agudizado nos últimos anos pela explosão de novos cursos de informática. Considerando a crescente demanda por mão de obra , e o conseqüente interesse crescente da comunidade de entrantes no terceiro grau, as universidades e faculdades tem expandido largamente a oferta de tais cursos.

Na cidade de Curitiba, que tem um porte médio , existem hoje mais de 30 cursos superiores diretamente relacionados à informática. Há 10 anos, este número era inferior a 10.

A nova legislação do ensino superior (Decreto Lei n. 2306 de 19/08/97) [2], facilitou a criação de nova universidades e centros universitários e propiciou a entrada de vastos contingentes humanos no terceiro grau. Esta situação tem o mérito indiscutível de alavancar a qualidade de vida das pessoas através da educação. Há indiscutível correlação entre os índices de desenvolvimento de uma sociedade e o número de pessoas portando diploma superior.

Um subproduto menor da situação acima descrito traz implicações para os que trabalham no ensino da informática. E, do nosso ponto de vista, implicações negativas diga-se. Trata-se da diminuição da relação candidato / vaga na entrada dos cursos universitários de informática. Embora eu desconheça estudo conclusivo, a prática mostra que quanto

menor é esta relação, menos qualificados são os alunos entrantes. Maiores, portanto, as dificuldades na apresentação e posterior apreensão do conceito de algoritmos.

Em nossa instituição, a Universidade Positivo (UP), há 10 anos, a relação candidato / vaga para o curso de informática era de 4 candidatos por vaga para os cursos noturnos e de um pouco mais de dois para um nos cursos diurnos. Esta relação se manteve mais ou menos constante por diversos anos, ainda que em um deles, tenhamos tido uma procura atípica que elevou os índices para 7:1 e para 4,5:1 respectivamente. Entretanto, nos últimos 3 anos, em nenhum vestibular, qualquer das relações foi superior a 1:1. Isto significa que nos últimos 3 anos, sempre houve mais vagas do que candidatos.

O impacto desse estado de coisas para o ensino de programação é significativo. Subitamente vimo-nos frente a grupos de alunos que entraram no curso sem outra credencial além da habilidade de não zerar nenhuma prova do vestibular. Seu desempenho em disciplinas do segundo grau, particularmente a matemática, irmã gêmea da programação, deixava a desejar.

Ainda introduzindo o assunto, cabe uma constatação: as práticas de ensino de informática tem usado muito pouco o potencial representado pelo computador na sala de aula. Afora as indefectíveis aulas de laboratório onde – em geral – um professor escreve algoritmos no quadro e os alunos copiam-no e tentam executá-lo, pouco esforço parece ter sido dispendido na busca de maneiras de pôr o computador a auxiliar o aluno e o professor nesta tarefa. Parece ser mais uma manifestação do adágio "casa de ferreiro, espeto de pau". Há falta de softwares, metodologias, iniciativas e relatos de experiências que usem a máquina para realmente ajudar a programar, afora o uso óbvio que é pela compilação e execução de exercícios.

Comecei a ensinar informática em 1976, e algoritmos e programação a partir de 1989. A partir de 1993 passei a lecionar também estruturas de dados, que nada mais são do que "algoritmos e programação"mais sofisticados. Conhecendo um pouco da capacidade e habilidade dessas máquinas, desde que programadas, comecei a pensar em como elas poderiam ajudar alunos e professor em atingirem seus alvos comuns.

Nasceu assim, a partir de 1992, a idéia dos algoritmos vivos. Ainda não tinham este nome, nem a estrutura conceitual que será a seguir apresentada, mas não importa: aquele exercício que pedia aos alunos para criar e usar uma árvore de Hufmann para codificar e comprimir uma pequena frase era a semente de algo que iria crescer e frutificar.

O que é um algoritmo vivo

Na visão de hoje, os algoritmos vivos são uma ferramenta de ajuda ao ensino e aprendizado da programação. Do ponto de vista do aluno, trata-se de uma folha de papel contendo três coisas:

- A teoria subjacente a um determinado tópico, incluindo-se os algoritmos necessários escritos em pseudo-código.
- Um exemplo completo do uso desses algoritmos, especificando-se os dados de entrada, o tratamento sofrido por esses dados e o resultados que eles produzem. Este exemplo é o mesmo para todos os alunos.
- Uma nova instância de execução, na qual apenas os dados de entrada são apresentados, pedindo-se ao aluno que, em seguida, produza o resultado que o algoritmo produziria se executado fosse em um computador. Esta nova instância tem dados diferentes para cada aluno. As respostas sempre são um conjunto pequeno de dados.

Cada aluno recebe uma folha, personalizada, com seu nome e demais informações que auxiliarão na correção automatizada do exercício. Obviamente, se cada aluno tem a sua folha, os dados que a compõe podem ser distintos. Relembre-se que esta folha é emitida por um programa de computador, que portanto, pode e gera dados distintos para cada aluno.

Do ponto de vista do professor, trata-se de ferramenta importante, já que permite que cada aluno se defronte com um problema diferente, impedindo aquele comportamento muitas vezes usual: um ou dois fazem os exercícios, enquanto os demais divagam, sendo o resultado obtido ao final devidamente socializado entre todos.

Ao liberar o professor de conduzir a iniciativa comum de solucionar a questão, o uso desta técnica permite que ele atenda os alunos, atacando as dificuldades de cada um de maneira individual. Permite também que cada aluno siga o seu ritmo e trabalhe na velocidade que lhe é peculiar, liberando-se a turma da necessidade de todos irem atrelados a um determinado ritmo e que em geral é determinado pelos alunos que maior dificuldade apresentam.

Certamente isto está de acordo com a moderna pedagogia que apregoa a necessidade de limitar aulas meramente expositivas. Tenho apresentado esta questão a meus alunos de maneira simples e direta: quem tem que trabalhar é o aluno. O professor, supostamente, já sabe isso tudo. A ele cabe ajudar os que tem dificuldade e não atrapalhar, saindo da frente dos que vão sozinhos. Essa idéia não é de minha autoria. Outros, mais abalizados já o disseram, como por exemplo, Lauro de Oliveira Lima: "o problema da aula expositiva é que, ou é intragável e suporífera ou é brilhante, cativante e dramática. No primeiro caso, simplesmente não existe. No segundo, o próprio brilhantismo do orador e o interesse em acompanhar a exposição, impedem qualquer reflexão, ..., de modo que terminada a aula, ficam

boiando na consciência, apenas alguns pontos pitorescos, quase sempre os menos relevantes". [4] Coube-me meditar sobre o assunto, admitir a veracidade do enunciado e trabalhar para acabar com as aulas só expositivas.

A possibilidade de realizar muitos trabalhos individuais ao longo do bimestre, tipicamente mais de 10, permite que sejam atribuídos graus a tais trabalhos, minimizando a importância, inclusive quantitativa, da prova bimestral. Assim, o conceito numérico do aluno vai sendo construído dia-a-dia, dando chance aos que apresentam maior dificuldade, orientando o estudo, diluindo-o ao longo do tempo e não apenas apresentando uma prova que dará origem a um grau absoluto e irrecorrível.

Na Universidade Positivo, onde os algoritmos vivos estão sendo usados em sua última encarnação, eles têm contribuído com 60 % da nota bimestral, reservando-se os 40 % restantes para a prova, que até mesmo por ter tido sua importância relativa diminuída, pode ainda continuar sendo de uma exigência formal e conceitual absoluta. Se toda a nota do bimestre corresse por conta apenas do exame, os critérios de correção teriam que ser relaxados, sob pena de médias baixas e conseqüente insucesso de grande parte dos alunos.

O programa que gera os algoritmos vivos, gera também as respostas dos mesmos. Afinal, se cada aluno tem um exercício diferente, há que se dar ao professor alguma facilidade na correção. Isto ocorre de tal maneira que é possível até, transferir a tarefa de correção a monitores ou outros auxiliares que não precisam conhecer o conteúdo ou o tema de estudo. Tal resposta é gerada em papel e também em mídia magnética, permitindo uma correção vis-a-vis com a resposta dada, ou alternativamente a digitação dos dados gerados pelo aluno. Neste caso, depois de corrigí-los, o programa gerador dos algoritmos vivos produz um arquivo que contém a identificação do aluno, a identificação do evento (neste caso um exercício e qual deles) e o grau obtido. Um outro software de minha autoria denominado PANA, que gerencia alunos, turmas, conteúdos, datas, provas, exercícios e notas captura tal arquivo e ao final do bimestre, calcula médias e as envia - por meio eletrônico - para a secretaria da instituição. Afinal, como se disse acima, quem tem que trabalhar é o computador.

Finalmente, como a emissão de uma coleção de exercícios individualizados é obtida apenas "pressionando-se alguns botões", o uso de algoritmos vivos permite variações interessantes na sua aplicação. Assim por exemplo, quando um resultado ruim é alcançado, o mesmo exercício pode ser refeito, várias vezes até. Não há problemas, já que a cada emissão, embora o enunciado permaneça o mesmo, os dados de entrada são distintos. Outra abordagem interessante é a criação de pequenos grupos, cada um com um exercício, para que ocorra o trabalho conjunto. Isto é usado em algoritmos vivos difíceis e/ou demorados.

Particularmente, eles tem sido usados nos últimos 2 anos, em uma modalidade que aparentou ser a melhor, depois de diversas tentativas, acertos e erros: Uma aplicação de um determinado algoritmo é feita, normalmente em sala de aula. Nesse dia, o resultado não é levado em consideração para a nota bimestral, e conseqüentemente os resultados esperados estão à disposição dos alunos que podem assim acompanhar seu desempenho e discutir com o professor acertos e erros. Esta primeira execução serve também para apresentar o exercício aos alunos. Em uma próxima aula, com alguns dias de intervalo (o que permite que interessados implementem e experimentem o algoritmo rodando em computador), uma nova emissão é apresentada. Nesta segunda oportunidade o exercício vale nota e o resultado não está disponível aos alunos. Estes o receberão apenas no futuro, já com o exercício devidamente corrigido e valorado.

Para encerrar esta apresentação, vale citar que a partir de 2001, o arcabouço conceitual dos algoritmos vivos foi usado para uma variante do software que gera provas. Aqui, a ênfase está em perguntas e resposas esperadas. A vantagem é a de permitir que cada aluno tenha uma prova diferente, sempre com questões de complexidade e temática similares, todas retiradas de um banco de questões especialmente preparado. Os resultados desta última iniciativa ainda não estão disponíveis, devido ao pouco tempo de uso.

Anatomia de um algoritmo vivo

Os algoritmos vivos estão 100 % programados em APL [3]. Embora tal linguagem não seja de uso disseminado, nem conte com muitos adeptos, ela definitivamente deixa qualquer outra atrás quando se trata de escrever programas exploratórios, complexos, com alto grau de manuseio nos dados de entrada. Perde, sem nenhuma dúvida, em desempenho, mas isto aqui não é importante. Sua interface é ruim, ela usa um alfabeto especial e é orientada a caracter, mas ultrapassado este desconforto, abre-se um paraíso ao programador. A linguagem parece ter sido construída ad hoc para os algoritmos vivos. Com ela é possível produzir um novo exercício em poucas horas. Desconfia-se que ao usar qualquer outra linguagem, seriam semanas e não horas.

Já estão disponíveis todos as funções que implementam as partes comuns dos algoritmos vivos. Para cada novo exercício é necessário criar apenas as seguintes funções específicas daquele exercício:

- Uma função - usualmente denominada GERACASO - que gera um caso novo, aleatório, mas que permite solução em tempo hábil (estimado em cerca de 1 hora de trabalho para o aluno).
- Um conjunto de funções - dependentes do problema - que recebem o caso acima gerado e o resolvem gerando o gabarito para o professor.

Catalogação

Para a construção dos algoritmos vivos criei um índice de catalogação, parecido ao em uso nas bibliotecas que permite numerar os algoritmos vivos e inserir cada um em um contexto amplo da ciência da computação. Os grandes temas desta classificação são:

1 generalidades
2 alocação encadeada e apontadores
3 listas, pilhas e filas
4 árvores
5 ordenação, estruturas tipo hashing e cadeias
6 grafos
7 tópicos avançados
8 outros
9 não categorizados anteriormente

Conclusões

1. As aulas de estruturas de dados e de tópicos avançados em informática tem sido mais participativas, produtivas e agradáveis. Alunos têm manifestado o interesse que a abordagem lhes desperta. Algoritmos que assustam pela complexidade quando são apresentados, revelam sua singeleza quando postos em execução. Ou, ao contrário, os detalhes dinâmicos de um algoritmo aparentemente simples, quando vivenciados pelos alunos, mostram sua complexidade.

2. O entendimento e a depuração de programas baseados nos algoritmos estudados fica facilitado, relatam os alunos. De posse de um algoritmo vivo preenchido e entendido é menos difícil projetar e construir um programa de computador que funcione.

3. As médias bimestrais das turmas onde os algoritmos vivos são usados têm crescido. A disciplina de estrutura de dados e arquivos, que há vários anos, tinha o duvidoso título de "a que mais reprova alunos", perdeu-o, e passou a ter índices similares, e em certos casos, melhores do que as demais disciplinas do curso.

4. A parte expositiva da matéria, ainda e sempre necessária, foi reduzida, já que algoritmos vivos consomem muito tempo em sala de aula. Ela precisou ser apurada, concentrada, despida de trivialidades. Desconfia-se que o que permaneceu, ficou enriquecido e fortalecido.

5. A perda de alunos ao longo do ano também diminuiu. Era comum a desistência de grandes contingentes de participantes. Alunos que aparentavam estarem "perdidos"iam desistindo, sobretudo no início do curso. Hoje, o número de desistentes é menor e é compatível com as demais disciplinas do curso.

6. Finalmente, os algoritmos vivos tem ajudado a planejar e dar versões mais enriquecidos das disciplinas de estruturas de dados e de tópicos avançados. Embora representem uma carga adicional de trabalho para o professor na preparação e correção, esta carga tem sido contrabalançada com os resultados positivos obtidos.

Bibliografia

1. De MARCO, Tom e LISTER, Timothy. Peopleware. Como Gerenciar Equipes e Projetos Tornando-os mais produtivos. Ed McGraw Hill. 1990.

2. DECRETO LEI n. 2306, de 19 de agosto de 1997, que regulamenta as Instituições de Ensino Superior no Brasil. Disponível na íntegra em www.mec.gov.br/Sesu/ftp/Decreto2306.doc.

3. KANTEK, Pedro. Uma Introdução ao APL. CELEPAR. 1982.

4. LIMA, Lauro de Oliveira. Treinamento em Dinâmica de Grupo. Vozes. 1973.

Apresentação

Esta metodologia foi apresentada em um congresso internacional de Pedagogia na área de informática e publicado nos anais, como se pode ver em *NAVARRO, P. L. K. G. . A Toolbox To Teaching And Learning Data Structures: Live Algorithms. In: International Conference on Education and Information Systems: Technologies and Applications (EISTA '03), 2003, Orlando, 2003.*.