基于学习者视角的算法可视化系统研究综述

李晓鸿 刘 丛 骆嘉伟
(湖南大学信息科学与工程学院 长沙 410082)

摘 要 算法理论复杂、概念抽象,通常给学习者带来一定的困扰。算法可视化通过将具体算法流程进行高层次抽象,并通过算法动画的形式展现出来,使算法过程形象可见,从而降低算法的理解难度,对于学习者来说具有重要的作用。目前已经存在许多算法可视化系统,学术界常以 Price 和 Karavirta 等人的分类方法对算法可视化系统进行讨论,但其以系统功能为标准的分类方法较为复杂,同时分类并非只针对算法可视化系统,不易于理解。以学习者的角度对经典的算法可视化系统进行重新分类,总结了算法可视化系统的历史现状,并讨论了算法可视化系统未来的发展方向。

关键词 算法可视化,算法可视化系统,算法学习,计算机辅助教学
中图法分类号 TP393 **文献标识码** A

Review of Algorithm Visualization Systems: A Learner Perspective

LI Xiao-hong LIU Cong LUO Jia-wei
(College of Information Science and Engineering, Hunan University, Changsha 410082, China)

Abstract Assisting students to understand algorithms is a challenge task in computer science education. Since algorithms are often complex topics, algorithm visualization is a useful aid for understanding the working of algorithms. It visualizes the behavior of an algorithm by producing an abstraction of both the data and the operations of the algorithm. Many algorithm visualization systems have been developed over the last years, Price and Karavirta's taxonomy classifies the features and characteristics of different visualization systems which are widely accepted. However, Price's taxonomy is too complex, and focuses on many irrelevant visualization systems which may confuse the readers. This paper gave a new taxonomy of traditional algorithm visualization systems from the learners' view which is easy to understand. Also, we summarized the history and current situation of algorithm visualization systems and discussed the future of them.

Keywords Algorithm visualization, Algorithm visualization systems, Algorithm learning, Computer assisted instruction

1 算法可视化系统背景介绍

算法可视化是将一段程序的数据、操作、语义进行抽象,并对这些抽象进行动态的图像展示[1]。在可视化技术发展初期,Price 等人将算法可视化归为软件可视化(Software Visualization)领域[2]。随着信息时代的到来,知识量爆增,并逐渐产生了知识可视化(Knowledge Visualization)的概念。知识可视化指的是所有可以用来建构和传达复杂知识的图解手段[3]。算法的实现与应用多以软件为载体,与软件有着密不可分的关系,同时算法概念与思想又是一种复杂的知识,与软件无关。因此,算法可视化应该属于软件可视化与知识可视化的交叉领域。

算法本身具有算法思想、算法流程等知识化的概念。算法可视化的一个任务就是将算法中所涉及的知识进行具象化,使之更容易理解、传播和交流。以数据结构中的"树"为例,树是一种非线性结构,而且通常具有层级性和递归性。在数学定义中,树可以通过广义表的形式进行表示,如($A(B(E,F(I,J)),C,D(G,H))$)(在广义表中,一个节点的子节点都写在该节点名称后的括号中,且具有嵌套的形式)。这种形式简洁,便于存储和传递,但不够直观,不便于理解和交流。通过算法可视化,可以将该树表示为如图 1 所示的形式。显然,经过可视化后的"树"更自然、直观、容易理解。

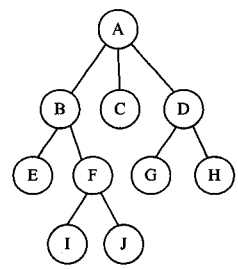


图 1 树的可视化形式

算法在计算机中体现为算法代码,而算法可视化的另一个任务就是将代码可视化为对应的算法动画。根据代码的可视化程度不同,又分为程序可视化与算法可视化[2],二者十分容易混淆。

程序可视化(Program Visualization)对程序中的语句、变

本文受 2012 年湖南省普通高等学校教学改革研究项目资助。

李晓鸿(1973—),男,博士,讲师,主要研究方向为算法可视化,E-mail:lixhong@263.net;**刘 丛**(1991—),男,硕士生,主要研究方向为算法可视化;**骆嘉伟**(1964—),女,博士,教授,博士生导师,主要研究方向为数据挖掘、生物信息处理。

量进行直接可视化。例如程序可视化通过显示某个循环的执行次数，来教会学习者如何正确写出循环的边界条件，或者通过表格的形式监视一些变量在程序执行过程中值的变化，来使他们掌握一些赋值语句是否起到了真正的效果。程序可视化通常会对每一条语句都进行可视化，通过程序可视化，学习者能够很快熟悉编程语言的语法规则，了解不同的语句的作用。图 2 为对一个通用 continue 语句所进行的程序可视化。

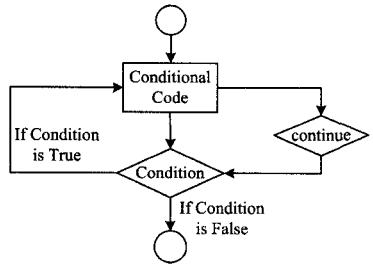
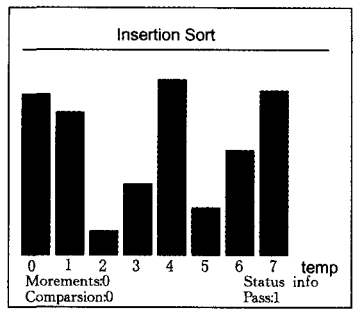


图 2　C 语言中 continue 语句的逻辑图




```
insertionSort(list)
1     for( index = 1; index <
  list.length; index++)
2     temp = list[index]
3     vacant = index
4     while( vacant not at start of
  list and there has benn a change)
5       if( list[vacant - 1] > temp)
6         swap(vacant, vacant - 1)
7       vacant - -
8     list[vacant] = temp
  Variables:
        index  = 1
        temp = -
        vacant = -
```

图 3　插入排序算法可视化

算法可视化(Algorithm Visualization)是将程序的代码、操作、语义进行抽象，并对这些抽象进行动态的图像展示。以排序算法为例，算法可视化系统首先会将程序中的代码进行组合抽象，将赋值和循环语句转化为排序算法中的元素交换、插入操作，而后在程序执行到这些语句时，系统会根据其对应的操作，展示排序算法中相对应的动画。算法可视化不会对每一条语句进行可视化，它关注的是语句整体所表达的逻辑、算法。通过算法可视化，学习者能够很快建立起算法与代码之间的联系，提高自己的编程能力。图 3 为插入排序的算法可视化截图。

2　对已有算法可视化系统分类分析

Price 等人曾对软件可视化系统进行研究并分类[2]，其分类方法得到了领域内的普遍认可。Price 等人认为不同的系统千差万别，但其框架都是相似的。例如对任何一个可视化系统来说，它都需要进行展示并与用户进行交互，不同的只是展示内容的方式和交互的方式不同，因此按照这种规则，可视化系统首先被划分为 6 大部分：类型(Scope)、可视化的内容(Content)、构成(Form)、可视化技术(Method)、交互(Interaction)、效率(Effectiveness)。其中每一部分又分为若干特征，例如构成包括了是否使用声音(sound)，是否使用色彩(colour)，是否使用动画(animation)等。软件可视化系统通过 6 大部分共计 30 多个特征进行评价和分类。表 1 列出了分类，其中最左列为不同的系统名，第一行为特征名，中间内容通过 Yes、No 或具体的文字来区分不同系统在不同特征上的差别。

表 1　Form 大类分类表[2]

C. Form	12 Medium	13 Graphical Elements	14 Colour	15 Animation	16 Multiple Views	17 Other Modalities
SOS	Film	Rectangles, lines and dots	Yes	Yes	No	Sound
BALSAII	Workstation	2D graphic primitives and text	Yes	Yes	Yes	Sound
SEE	Paper	Multi-font typeset text, rules and grey tones	No	No	No	None
M & S	Workstation	2D graphic primitives and text	No	Yes	Yes	None
TPM	Workstation	2. 5D graphic primitives and text	Yes	Yes	Yes	None
Logo Motion	Workstation	2D graphic primitives and text	No	Yes	Yes	None
UWPI	Workstation	2D graphic primitives and text	No	Yes	No	None

Price 等人的分类方法对算法可视化领域做出了重要的贡献，但其分类方法仍有以下两方面的不足。

1)其分类方法过于复杂。Price 等人的工作更像是对不同的系统进行详细的描述而非分类，通过几十种属性来定义一个系统的方法并不利于对算法可视化系统产生一个整体的认识。

2)其分类方法并非只针对算法可视化系统。算法可视化系统最早被归为软件可视化系统领域，其分类方法包含了许多非算法可视化系统，因此一些属性对于算法可视化来说是不必要的。同时，经过十几年的发展，算法可视化已经逐渐完善成熟，并具有自己的特色，足够脱离软件可视化系统进行单独的研究与讨论。

M H. Brown[4] 等人将与算法可视化系统相关的人员分为开发者(Developer)与使用者(User)。开发者是指开发算法可视化系统的人使用者又分为 4 种角色：算法设计者(Algorithm Designer)，希望通过算法可视化系统能让程序变得更容易理解的一类角色；动画师(Animator)，为程序设计对应的动画的一类角色；脚本师(Scriptwriter)，编写脚本，调用系统封装动画的一类角色。学习者(Learner)，观看算法可视化动画的一类角色。

算法可视化在教育领域、软件调试和思维启发方面都能发挥不错的作用，但事实证明算法可视化在后两者上面还有很长的路要走，算法可视化目前仍然最适合于教育领域[5]，然而 Price 的分类更多的是针对开发者而非使用者。对于使用者而言，也希望对算法可视化系统有一个整体的、系统的认识，从而能够根据自己的需求，选择合适的算法可视化系统。对于使用者，尤其是学习者而言，他们更关心的是系统的教学内容和是否适合自己使用。因此本文以学习者的角度对算法可视化系统进行重新分类，将传统的算法可视化系统归为：算法演示类、伪代码演示类、代码演示类、算法语言类以及算法工具类 5 大类，并主要评价系统所发挥的教育作用。

3　基于学习者视角的算法可视化分类

3.1　算法演示类

算法演示类是以算法思想、流程为主要内容，通过让学习者自由控制算法动画的播放速度、流程，以及所演示算法的数

据，使学习者更容易掌握算法的基本思想、熟悉算法流程的一类系统。

Sorting out sorting[6]（Sos）作为大家公认的最早的算法可视化系统，就是一个典型的算法演示类系统。Sos 系统由多伦多大学的学者（在 1979～1981 年）开发，他们利用大型工作站制作了一个 30 分钟的教学电影，该电影首先介绍了 9 种不同的排序算法，而后将 9 种不同算法运行于独立的并行系统上，通过进行一个排序速度竞赛来对比 9 个排序方法的效率，并最终给出各个算法的效率分析。图 4 为 Sos 电影排序竞赛片段的截图，在电影开始时，所有的数据都是随机分布，在电影中显示为随机分布的点，随着排序进行，数据逐渐呈现有序性，逐渐接近一条斜线，不同排序算法效率不同，因此在同一时间，各个算法的有序性也不一样

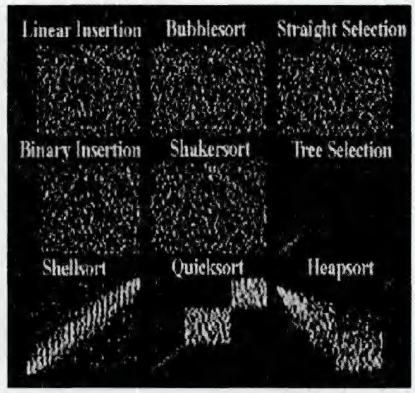


图 4　Sos 系统排序电影[6]

Xtango[7] 系统是 Tango[1] 系统的一个衍生版本，二者皆由布朗大学的学者开发。Tango 系列出名是因为它提出了一个路径转换范式和一个算法可视化系统的开发框架，使得动画生成更容易。但从 Xtango 系统最终使用效果来说，它仍是偏重于算法演示类的系统。图 5 为 Xtango 系统关于数据结构“堆”的演示，在最初，屏幕只有一个数据，在点击界面左侧插入（in）数据按钮后，系统右侧会展示新加入一个元素的动画，删除也类似，图为经过若干次操作后的结果。

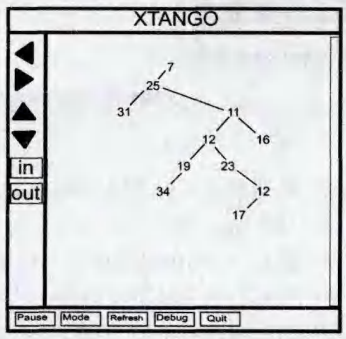


图 5　Xtango 堆演示[7]

Javanga[8] 系统是基于 Java 语言开发的专门针对图和网络算法的可视化平台，它不仅提供了图的一般算法演示，还包含了注入单纯形法、网络流等复杂的问题。它界面精简，具有较好的交互性，能快速创建一个算法案例并观看算法动画。同时系统还会将必要的信息，例如算法的概念介绍、执行过程中关键变量的信息，显示在专门的信息区域来辅助用户更好地学习算法。图 6 为 Javanga 系统最小生成树算法演示截图，与图 5 的 Xtango 类似，Javanga 系统通过最上方的按钮控制动画流程，在图结构发生变化过程中数据的变化通过右侧窗口进行输出跟踪，。

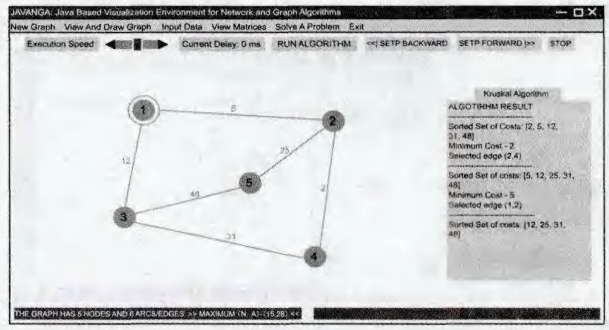


图 6　Javanga 克鲁斯卡尔最小生成树算法演示[8]

在算法学习过程中，首先是了解算法思想，熟悉算法流程，由于这时的学习不涉及代码，因此算法演示类系统的主要工作是算法流程的可视化展示，并辅佐关键变量的可视化跟踪。算法演示类系统能够让学习者快速地对某个算法有一个整体的认识，熟悉算法的特性和思想，统非常适合初学算法的人使用。同时，精通算法的人仍然可以使用算法演示类系统快速构建一个算法流程用于检验，或得到结果。

3.2　伪代码类

伪代码类系统是以帮助学习者更容易掌握算法实现为目的的一类系统，它通常将算法的伪代码和算法的动画关联起来，通过将正在执行的代码行进行高亮，并与动画关联，表现出一种执行代码时演示算法的效果。

早期的伪代码类系统中典型的是由布朗大学开发的 Balsa[4] 系统。如图 7 所示，左侧为编译原理中语法分析算法的代码，右侧为一个可视化的语法分析树，上方为一个实际的例子。通过高亮的形式，系统将左侧的代码与右侧的动画关联起来，展示了算法针对一个实际表达式建立语法分析树的过程。

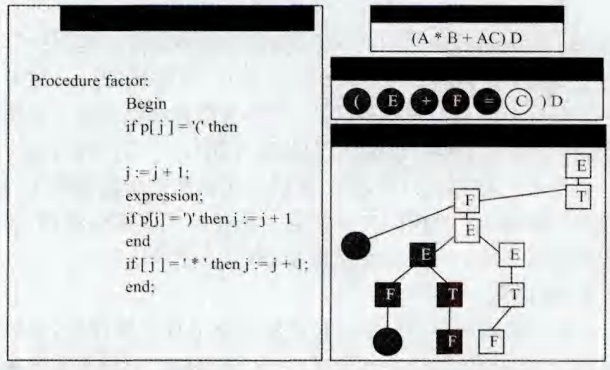


图 7　Balsa[4] 系统关于语法分析树构建的算法演示

JHAVÉ[9,10] 系统采用 JAVA 开发，使用者可以在互联网上直接使用 JHAVÉ 或者下载到个人电脑上作为独立应用使用。JHAVÉ 系统内置了许多算法供用户选择观看，包括常见的排序算法和一些典型的数据结构算法。JHAVÉ 支持自由输入数据或由系统生成随机的数据；同时还支持“弹出式问题”[11]，现在许多开发者都将该功能集成到了算法可视化系统中。弹出式问题即在算法演示过程中弹出一个对话框随机询问观看者一些问题，例如下一步某个变量值会变成多少，用

户输入正确的结果后，算法演示继续执行。弹出式问题从某种程度上能够让用户参与系统的程度更高。图 8 为 JHAVÉ 系统演示界面。

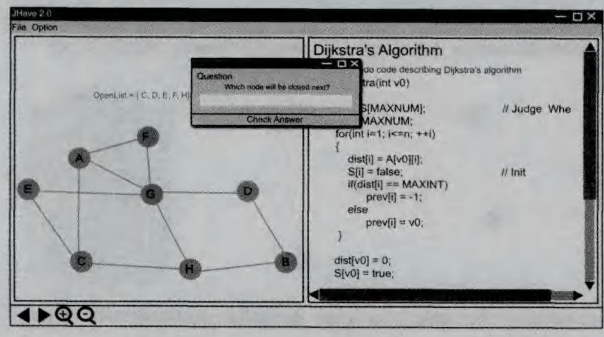


图 8　JHAVÉ[9]中狄杰克斯拉算法的演示界面和弹出式问题截图

AnimalSense[12]同样作为代码类系统，具有该类系统一般的所有特性。同时 AnimalSense 内置了一套练习评估系统，它内置许多问题模板，并通过改变模板的参数、数值来不断生成不同的问题以考察学生是否真正掌握了某个算法，其开发者认为，提出一些有意义、具有挑战性的问题，将对学生十分有帮助。图 9 为 AnimalSense 系统算法演示及互动问题截图，左为选择排序算法演示，右为其互动练习界面[12]。

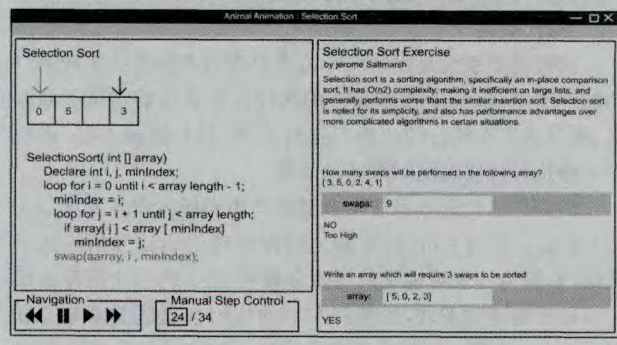


图 9　AnimalSense 算法演示及问题截图

伪代码类系统通过将代码与算法动画相关联，让学习者加深了代码与算法之间的联系，使得学习者更容易地在计算机上实现某个算法。在伪代码类系统中，其展示的代码并不一定是完整的可以执行的代码，也可以是具有象征意义的伪代码。事实上，这些代码并不是真正在执行，只是开发者通过一些技术[13]模拟执行的效果，所以伪代码类系统提供的代码通常是固定的。伪代码类系统对于已经熟悉了算法特性、流程并想要开始学习实际编程的人来说十分适合。

3.3　代码类

代码类系统是需要用户自己编写部分或完整代码，而后将用户所编写的代码进行可视化的一类系统。代码类系统通常会以某种编程语言（例如 C 语言）的语法规则为基础，在完成一个 C 语言编译功能的基础上，还能将可视化元素与使用者所写代码中的语句进行绑定，在代码执行到相关语句的时候，提供对代码的可视化功能，从而提高使用者的算法实现和应用能力。

如华盛顿大学的学者开发的 UWPI[14]系统，首先使用者需要在代码区写出一个算法的完整源代码，如图 10 所示，左侧为使用者自己写的图的广度优先遍历算法函数，由于函数中含有二维数组参数，系统会自动（或由使用者手动指定）将该二维数组识别为图的邻接矩阵。而后函数中对该二维数组的操作都会被转化为对图的操作并更新在动画上。系统通过自己的识别算法对代码进行分析，猜测可能会使用到的算法或数据结构，这种识别不是精确的，因此系统会为每种可能的算法或数据结构提供一个权值，权值较高的就作为代码的高层次抽象并以此进行可视化。

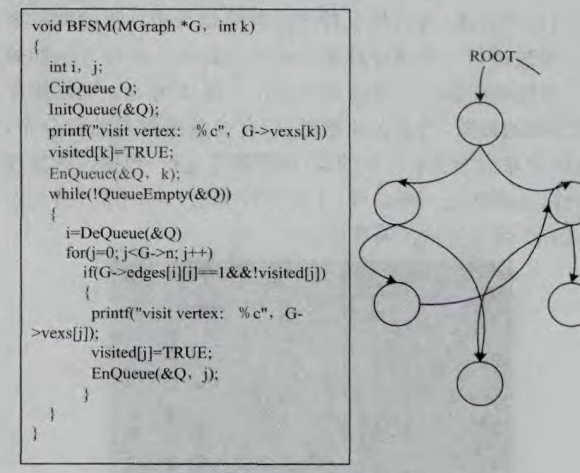


图 10　UWPI 系统[14]

Leonardo[15]是一个基于 C 语言语法的可视化系统，系统构建了一个虚拟的 CPU 用来模拟代码的执行。该系统的一个特点是允许程序进行逆执行，系统需要通过一个声明语言来完成实际代码和可视化元素的映射，但是，Leonardo 系统只能够运行在 MAC 和 PowerPC 处理器上。图 11 为 Leonardo 系统界面截图，左图为代码编辑界面，右图为左图代码的可视化。

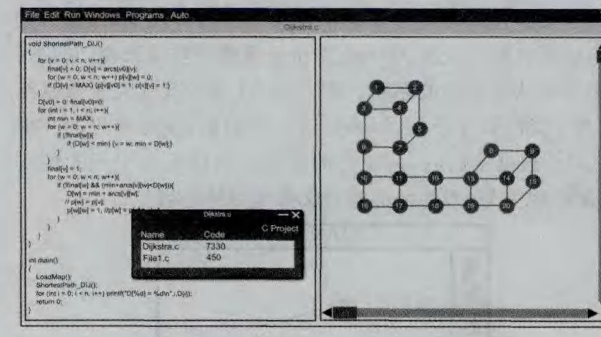


图 11　Leonardo 系统[15]

算法和数据结构通常会用到不同的基础类型，比如数组、指针。在伪代码中这些基础类型会以抽象的符号进行表示，例如数组用 array 表示。而伪代码翻译到具体的不同编程语言仍然会有许多的差别。首先是不同的语言对同一基础数据类型的声明和使用方式可能会不同；其次，并不是所有编程语言都具有某些特性，比如 Java 语言里就没有指针的概念。由于代码类系统在代码上为使用者提供了更多的自由性，因此使用者不仅能够通过系统进一步训练运用代码实现和应用算法的能力，提高编程能力，还能通过可视化来检测自己的代码或逻辑是否正确。代码类系统很适合想要进一步锻炼算法实现和应用能力的使用者。

3.4　语言类

算法可视化语言（algorithm animation language），是由手

工编写或程序生成，并能在一个或多个算法可视化系统翻译为对应算法动画的一种描述型语言。严格意义上来说算法可视化语言并不是一种系统，但仍然将其作为一种分类讨论。

在算法可视化系统发展初期，算法的可视化动画都集成在算法可视化系统里，使用者只能通过系统观看固定的可视化动画。后来开发者们将系统中的底层动画封装，并为使用者提供了调用这些动画的方式，即所谓的脚本。通过组合不同的脚本，使用者能够构建一些更复杂的动画。但系统千差万别，一个系统中导出的脚本可能无法导入到另一个系统中，系统之间的不兼容和不一致给使用者带来了诸多不便。同时，脚本的语法规则也因系统而异，很多系统需要书写复杂的脚本才能得到一个令人满意的算法可视化动画。因此随着算法可视化系统发展的逐渐成熟，许多研究者开始致力于这些脚本的研究，他们设计出了语法规则更完善的脚本系统，而这些脚本就是算法可视化语言。

Jawaa[16]是一种基于命令式算法可视化语言，用户可以通过简单的命令绘制基本的图形，如圆、多边形和简单文本，并能够通过功能命令改变这些基本图形的属性，如颜色、字体、位置、大小等。同时，Jawaa 还封装了许多数据结构，如栈和队列，使用者只需要调用一些基本的 Push 或 Pop 命令，就可以达到实际的入栈、出栈可视化效果。Jawaa 解释器采用 Java 开发，其动画可以方便地嵌入网页进行传播，其解释器具有暂停、连续执行动画脚本和脚本的逐条执行等功能。图 12 为 Jawaa 制作的 DFS 动画及其部分代码。

```
begin
node 1 120 30 20 10 1 "1" black lightGray black CIRCLE
node 2 70 120 20 20 1 "2" black lightGray black CIRCLE
end
connectNodes a 2 1 blck 0
Text post 250 60 "Postorder:" black
stack s 300 150 0 black red
push s 1
changeParam 1 bkgrd green
text t1 315 30 "1" green
moveMarker 20 2
text t14 420 60 "1" blue
delete 20
```

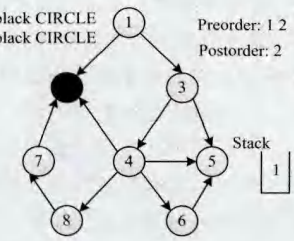


图 12　DFS 部分代码及其动画[16]

Xaal[17]意为可扩展性算法可视化语言(Extensible algorithm animation language)，它是一种 XML 语言，其开发初衷是为不同算法可视化语言提供一个转换的工具，因此 Xaal 语言被定位为一种通用的算法动画描述语言。它采用 XML 的语法规则，用不同的标签定义不同的算法元素及可视化行为，不同的系统能够根据标签将算法动画转化为该系统可识别的脚本。图 13 为 Xaal 语言定义的图样例。

```
<graph>
  <node id="n1"><key value = "D" /></node>
  <node id="n2"><key value = "E" /></node>
  <node id="n3"><key value = "M" /></node>
  <node id="n4"><key value = "O" /></node>
  <edge from="n1" to="n2" directed="true"/>
  <edge from="n2" to="n3" directed="true"/>
  <edge from="n3" to="n4" directed="true"/>
  <edge from="n4" to="n1" directed="true"/>
</graph>
```

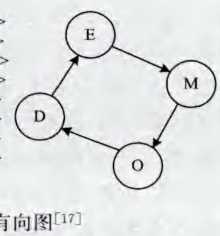


图 13　Xaal 定义的有向图[17]

Animalscript2[18]语言的语法规则更像是一门编程语言，相对于它的前一个版本 Animalscript[19]，二代版本支持循环、判断以及变量等编程语言中的特性以简化需要频繁操作的类似语句，从而让其显得更智能、易用。图 14 为 Animalscript2 所做的冒泡排序动画脚本。

```
array "values" (10 10) length 5 int "3" "8" "2" "1" "7"
int i = 4;
arrayMarker "i" on "values" atIndex i label "i"
int j = 1;
arrayMarker "j" on "values" atIndex j label "j"
for(; i>=0; i--){
  moveMarker "i" to position i within 5 ticks
  for(; j<=i; j++){
    moveMarker "j" to position j within 5 ticks
    if (values[j-1] > a[j])
      arraySwap on "values" position i with j within 5 ticks
  }
}
```

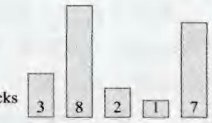


图 14　冒泡排序动画脚本[18]

算法可视化语言将算法可视化系统的开发与使用隔离开来。脚本设计师可以通过算法可视化语言来设计动画，学习者可以通过观看动画来学习。而开发者则需要根据语法规则去设计解析系统，事实上，许多算法语言的提出者都为之提供了一个解析器，但算法可视化语言并不都有或需要对应的解析器，算法语言更多地是作为一种描述算法动画的标准，便于交流和传播。算法可视化语言非常适合于教育者，通过算法可视化语言，教师能够很容易地设计出生动的教学案例，从而提高教学质量。

3.5　工具类

除了上述几种可以直接使用的算法可视化系统外，还有许多辅助算法可视化创作的工具。广义上讲，任何与算法可视化直接或间接相关的系统或工具，都可以称作算法可视化工具类，比如 Microsoft 公司的 Power Point(PPT)，和 Adobe 公司的 Flash，这也是目前教育者制作算法动画使用最多的工具。但我们要设计并实现与算法可视化直接相关的专业工具，并在一定程度上能够降低算法可视化的制作难度。

GeoWin[20]是一个专门为几何学算法可视化开发的一个工具，它使用 C++将一些底层动画进行封装，用户可以使用 GeoWin 封装好的一些接口进行二次开发，从而方便地创造算法可视化动画，图 15 为其截图。

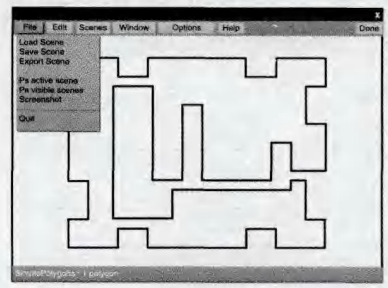


图 15　GeoWin 编辑多边形界面[20]

JSAV[21](JavaScript Algorithm Visualization) Library 是基于 JavaScript 语言的专门用于创作算法可视化系统的类库。它封装了一些数据结构可视化的元素与常见算法可视化过程，并将这些函数打包成库。其他开发者能够调用这些封装好的类库进行二次开发，从而能够快速有效地创建新的算法可视化系统，利用 JSAV 能够较为容易地制作算法可视化系统并以 HTML5 形式发布在互联网上，开发者无需再为每一个动画细节进行编程，只需要集中精力来处理系统的核心逻辑，其它的工作则交给 JSAV 处理。

算法可视化语言为动画制作者提供了便利，而算法可视化工具类则更多地针对算法可视化系统开发者。开发人员可

以方便地利用专用的算法可视化工具开发出更多新的、更易用的算法可视化系统。

4 算法可视化系统分析与评价

本文虽将算法可视化系统分为不同的5大类，但这5类系统却具有一定的联系。5类系统对于使用者知识水平要求具有递进关系。对于一个初学算法的人，其首先会通过演示类系统学习算法的基本思想与流程；在具有一定理论基础后，会通过伪代码类系统学习算法的实现；而后通过代码类系统练习编程与具体应用算法；最后还可以通过算法语言向别人演示算法甚至自己利用算法工具开发一个新的算法演示系统。在使用对象上，不同类的系统有不同的合适对象。演示类、伪代码类以及代码类更适合于学习者，语言类与工具类更适合于教育者(包括脚本师、算法设计者等角色)。文献[22]将算法可视化系统用户的参与度分为了6个层次，分别是：

第一级：无可视化(no viewing)，系统只提供文字、简单图表等无可视化的资料。

第二级：有可视化(viewing)，系统具有可视化，但用户无法与之进行交互。

第三级：反应(responding)，系统通过弹出式问题等与用户进行交互。

第四级：改变(changing)，用户能够改变系统所演示算法的数据等。

第五级：构建(constructing)，用户能够创建自己的算法可视化动画。

第六级：展示(presenting)，用户能够通过系统向他人展示一个算法。

虽然理论上一个系统所具有的层级越高，系统的使用效率也会越高，但对于使用者能力的要求也越来越严格。可以看出，本文所划分的5类系统在参与度的层级上也是递进的。图16为5类系统的关系图，图中矩形覆盖范围代表了系统所具有的可能性，不代表该类系统中的每个系统都具有同样层级。

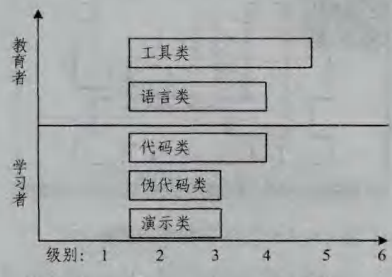


图16 系统关系图

算法可视化是伴随着需求而发展的一门技术，用户的需求决定了算法可视化的发展方向。而随着用户需求的不断提高与计算机的不断发展，传统的算法可视化系统在如今的环境下存在着以下4点问题：

1)系统"教学性"不足

在算法可视化发展初期，算法可视化系统的使用者通常都是已经在课堂上学习了一定基础知识的学生，算法可视化系统通常只是作为课堂的辅佐工具，用以加强或测试学生的算法学习。而近年来移动教育、SPOC、MOOC[23]等新的教学方式的出现，使得教学环境不再仅限于课堂，学生可以随时随地学习。而若要通过算法可视化系统来进行系统学习，前提是学习者在使用系统之前对所演示的算法有大概的了解，学习者如果是一个对算法一无所知的人，那么看到的动画不过是一堆图形而已，无任何意义。因此传统的算法可视化系统在如今环境下很显然存在着"教学性"不足的问题，不能满足个性化的自学需求。

2)系统交互不适当

无论是算法演示系统还是代码类系统，其交互都是用户与动画之间的交互，而不是用户与算法之间的。用户能够通过系统提供的按钮控制算法动画的播放速度，却不能与其中的内容进行交互，就像一个学生反复观看一个老师录制好的教学视频一样，理想的交互应该是能够和老师直接对话，所以目前算法可视化系统的交互功能仍然显得不够"智能"。其次，随着如今移动设备的流行，算法可视化系统的运行环境也发生了变化，而传统PC上的"键盘＋鼠标"的交互手段显然不适合移动设备。全新的交互内容和交互方式有待发现。

3)教学理念落后

2006年卡内基梅隆大学计算机系主任周以真教授提出了计算思维[24]的概念，并提出计算机专业的学生的能力不应该只是会使用计算机解决问题，而应当是像计算机科学家一样思考。显然，计算思维的提出对于计算机领域学习者提出了更高的要求，学生不仅需要知其然，更要知其所以然。而目前算法演示类系统的教学观念仍然停留在"教"会学生算法概念、使用算法等。学习者通过算法可视化系统掌握了算法知识，却不会变通与灵活应用，掌握了表面，却无法发现问题的本质。如何通过系统来训练学生的计算思维仍是一个难题。

4)系统实用性验证问题

算法可视化系统是否具有实用性并不是一个新问题，早在1996年，就有学者对算法可视化系统是否真的能够帮助人们更好地理解算法提出了质疑[25]。因此，算法可视化系统的开发者通常会对自己的系统进行实验。而实验内容通常是将学生分为两组，一组采用普通的教学方法，另一组配合使用算法可视化系统，而后通过对比学习的效果来得出算法可视化系统是否有效的结论。文献[26,27]对算法可视化系统的有效性进行了总结与评价。而实验的结果受多方因素的制约，如使用者、使用环境、学生参与度。例如学生的参与度越强，算法可视化系统的作用也相对越大。因此，算法可视化的实用性并不容易通过一些设计好的科学实验进行定性、定量的验证。首先，算法可视化没有一个标准，算法可视化的目标是什么，不同的开发者仍然有不同的想法，因此不同的算法可视化系统在其功能上会有所侧重，实验只能证明算法可视化系统在某些方面是否适用；其次，算法可视化系统实验受诸多系统外的因素影响，比如学生的理解能力、系统的使用环境、使用层次等，由于系统很难像算法一样定义一组输入、输出，在学生与可视化系统多层次交互之中，算法可视化实用性实验还需要更多的研究与设计。

5 未来工作

教学环境以及教学理念的改变给算法可视化带来了新的机遇与挑战。经过总结，我们认为算法可视化可以主要从以

下几个方面进行改进。

1)在内容设计上,将算法可视化系统设计为集学、练、测为一体的教学平台。如今随着算法可视化系统使用环境和使用对象的改变,算法可视化系统不再只是作为一个辅助的工具,特别是对于通过网络方式进行自学的学习者。学习者不仅可以通过算法可视化系统进行算法与数据结构概念理论的学习,还能够通过系统进行练习、测试。将算法可视化系统设计为一个具有完整功能的教学平台不仅能够给学习者提供一个完整的学习规划与流程,同时在一个平台下的教学内容、练习、测试能够更有针对性地关联起来,从而提高学习者的学习效率。

2)在交互的设计上,选择易用和参与度兼顾的交互方式。算法可视化系统在交互上首先应当具有易用性。在未来,算法可视化系统不能要求所有学习者在使用之前都进行过系统的使用培训,因此算法可视化系统在交互上应该设计得简单明了,学习者能够很快上手,从而不会将一部分精力花费在学习系统的使用上。其次,在满足易用的前提下,算法可视化系统还应该为学习者提供更自由的参与空间。例如给学习者提供简单的命令来调用不同的算法、改变算法中的不同数据。学习者不只是被动地观看学习算法,而是主动地参与到算法的执行与构建之中,从而使算法学习更有效率。

3)在教学理念选择上,以思维启发为中心。传统的算法可视化系统将教学核心放在了算法概念、代码实现上。然而在计算思维训的理念下,计算机只是一个工具,能够通过计算机编程实现算法并不等于能够通过算法思维解决问题。因此系统设计应该着重在算法性能、适用环境等高级理论上,而不应该将过多的精力放在代码实现等细节问题上。例如系统可以提供一些具体问题和一些封装了底层实现的算法与数据结构,学习者可以直接选择与组合不同的数据结构与算法来解决问题,通过提供不同方案的效率与结果,来让用户在使用和对比中掌握算法的精髓,训练计算思维。

4)将系统评价与系统相结合。毫无疑问,未来众多的互联网学习者将为算法可视化系统提供大量的实验数据。开发者可以在系统中记录学习者的学习行为以及评测结果。例如系统可以记录学习者在每个知识点的学习时间和测试的结果,来判断系统是否可能在某个知识点上讲解不够。然而由于互联网学习者的差异性较大,开发者还有许多额外的工作要做,例如在学习者第一次使用系统时通过问卷调查的形式来对不同的学习者进行分类,从而使实验结果具有更好的对比性和可信性。

除此之外,算法可视化系统还有许多其他工作可以开展。例如在移动设备上通过游戏的方式设计算法可视化系统,来解决移动设备学习时间碎片化和非精神集中状态下的算法学习问题。

结束语 算法可视化系统发展至今已有近50年的历史,而如今在新的教学环境、教学理念下,算法可视化也应不断地改进、改变以适应新时代下的需求。算法可视化还有许多问题值得讨论和研究,例如教育领域外,算法可视化究竟能发挥怎样的作用?相信只要众多学者一起努力,一定会做出更好的研究成果。

参 考 文 献

[1] Stasko J T. TANGO: A Framework and System for Algorithm Animation[J]. Computer, 1990, 23(9): 27-39

[2] Price B A, Baecker R, Small I. A Principled Taxonomy of Software Visualization[J]. Journal of Visual Languages and Computing, 1993, 4(3): 211-266

[3] Eppler M J, Bukard R A. Knowledge Visualization: Towards a New Discipline and its Fields of Application[D]. Lugano: University of Lugano, 2004

[4] Brown M H, Sedgewick R. A System for Algorithm Animation [C]// Proceedings of ACM SIGGRAPH'84. Minneapolis, MN, 1984

[5] Karavirta V, Korhonen A, Malmi L. Taxonomy of algorithm animation languages[C]// Proceedings of the 2006 ACM Symposium on Software Visualization (SoftVis'06). New York, NY, USA, ACM Press, 2006: 77-85

[6] Baecker R. Sorting Out Sorting: A Case Study of Software Visualization for Teaching Computer Science [M] // Stasko J, Domingue J, Brown M H, et al., eds. Software Visualization: Programming as a Multimedia Experience, chapter 24. MIT Press, Cambridge, MA, 1998: 369-381

[7] Stasko J T. Smooth continuous animation for portraying algorithms and processes[M]// Software Visualization. MIT Press, Cambridge, MA, 1998: 103-118

[8] Baloukas T. Javenga: Java-Based Visualization Environment for Network and Graph Algorithms[J]. Computer Applications in Engineering Education, 2012, 20(2): 255-268

[9] Naps T, Eagan J, Norton L. JHAVE—An environment to actively engage students in Web-based algorithm visualizations [J]. ACM SIGCSE Bull, 2000, 32(1): 109-113

[10] Naps T. Jhave: Supporting algorithm visualization [J]. IEEE Comput Graphics Appl, 2005(25): 49-55

[11] Karavirta V, Korhonen A. Automatic tutoring question generation during algorithm simulation[C]// Proceedings of the 6th Baltic Sea Conference on Computing Education Research. Koli Calling, Koli, Joensuu, Finland, 2006: 95-100

[12] Rößling G, Mihail M, et al. AnimalSense: Combining Automated Exercise Evaluations with Algorithm Animations[C]// ITiCSE 2011. Darmstadt, Germany, 2011: 27-29

[13] Brown M H, Hershberger J. Fundamental Techniques for Algorithm Animation Displays[M]// Stasko J T, Domingue J, Brown M H, et al., eds. Software Visualization. MIT Press, 1998

[14] Henry R R, Whaley K M, Forstall B. Common Lisp/CLX source code for an automatic Pascal algorithm animation system running on Unix workstations[EB/OL]. http://june.cs.washington.edu as/pub/uwpi.tar.Z

[15] Reversible Execution and Visualization of Programs with LEONARDO[J]. Original Research Article Journal of Visual Languages & Computing, 2000, 11(2): 125-150

[16] Akingbade A, Finley T, Jackson D, et al. JAWAA: easy web-based animation from CS0 to advanced CS courses[C]// Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education (SIGCSE'03). ACM Press, 2003: 162-166

(下转第453页)

著性差异的。所以，4.1 节实验所得结论在 95%的置信度是可靠的。

表 4 Kruskal-Wallis 检验结果

	秩		
	group	N	秩均值
CAS	1	10	14.70
	2	10	17.20
	3	10	35.20
	4	10	14.90
	总数	40	
ISBN	1	10	10.90
	2	10	30.50
	3	10	30.50
	4	10	10.10
	总数	40	
QUEEN	1	10	10.10
	2	10	30.50
	3	10	30.50
	4	10	10.90
	总数	40	
TRIANGLE	1	10	13.60
	2	10	30.50
	3	10	30.50
	4	10	7.40
	总数	40	

注：算法 1，2，3，4 分别代表 IBEA-HA、MOEAD-II、RANDOM、SPEA2+SDE

表 5 Kruskal-Wallis 检验统计量

程序	CAS	ISBN	QUEEN	TRIANGLE
卡方	21.376	33.467	33.471	35.047
df	3	3	3	3
渐近显著性	8.7928E−5	2.5666E−7	2.5621E−7	1.1905E−7

结束语 本文主要从进化算法在测试用例的自动生成方面的应用展开研究，以分支覆盖率和执行时间为双目标，研究了进化算法与随机算法以及不同进化算法之间在测试用例自动生成方面的执行效率，从而得出了在测试用例的生成应用中，进化算法的执行效率明显优于随机算法。特别地，SPEA2+SDE 进化算法的执行效率明显优于 IBEA-HV、MOEAD-II 两种进化算法。最后，为了说明实验结果的可靠性和广泛性，对上述实验结果进行 Kruskal-Wallis 检验，得出：实验结果所得结论在 95%的置信度是可靠的。

参 考 文 献

[1] Ferrer J, Chicano F, Alba E. Evolutionary Algorithms for the Multi-Objective Test Data Generation Problem [J]. Software: Practice and Experience, 2012, 42(11): 1331-1362

[2] McMinn P. search-based software test data generation: a suvery [J]. Software Testing, Verification and Reliability, 2004, 14(2): 105-156

[3] Li Mi-qing, Yang Sheng-xiang, Liu Xiao-hui. Shift-Based Density Estimation for Pareto-Based Algorithms in Many-Objective Optimization[J]IEEE Transactions on Evolutionary Computatin, 2014, 18(3)

[4] 韩丽霞. 求解多目标优化问题的新遗传算法[J]. 计算机科学, 2013, 40(6A): 64-66, 95

[5] 王静龙，梁小筠. 非参数统计分析[M]. 北京：高等教育出版社, 2006: 78-156

[6] Zitzler E, Laumanns M, Thiele L. SPEA2: improving the strength Pareto evolutionary algorithm: CH-8092[R]. Zurich, Switzerland, 2001

[7] Harman M, McMinn P. A Theoretical Empirical Study of Search-Based Testing: Local, Global, and Hybrid Search [J]. IEEE Transactions on Software Engineering, 2010, 36(2)

[8] Mark H, Kiran L, Phil M. A multi-objective approach to search-based test data generation[C]// Proceedings of Genetic and Evolution ary Computation(GECCO 2007). London, England, UK, 2007

[9] Mark H, Phil M, de Souza Jerffeson T, et al. Search Based Software Engineering: Techniques, Taxonomy, Tutorial[M]// Empirical Software Engineering and Verification: International Summer Schools, LASER 2008-2010, Elba Island, Itatly, Revised Tutorial Lectures. 2012: 1-59

[10] 杜强，贾丽艳. SPSS 统计分析从入门到精通[M]. 北京人民邮电出版社, 2009: 118-138

(上接第 437 页)

[17] Karavirta V. XAAL-extensible algorithm animation language [D]. Finland: Helsinki University of Technology, 2005

[18] Roβling G, Gliesche F, Jajeh T, et al. Enhanced expressiveness in scripting using Animal Script[C]// Proceedings of the 3rd Program Visualization Workshop. UK, 2004: 10-17

[19] Roβling G, Schüler M, Freisleben B. The ANIMAL algorithm animation tool[C]// Proceedings of the 5th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education(ITiCSE'00). Helsinki, Finland: ACM Press, 2000: 37-40

[20] Bäsken M, Näher S. GeoWin-A Generic Tool for Interactive Visualization of Geometric Algorithms[M]// Diehl S, ed. Software Visualization: International Seminar. Dagstuhl, Germany: Springer, 2001: 88-100

[21] Karavirta V, Shaffer C A. JSAV: The JavaScript Algorithm Visualization Library[C]// Proceedings of the 18th Annual Conference on Innovation and Technology in Computer Science Education(ITiCSE 2013). Canterbury, UK, 2013

[22] Naps T, Roβling G, Almstrum V, et al. Exploring the role of visualization and engagement in computer science education[J]. ACM SIGCSE Bull, 2003, (35): 131-152

[23] 樊文强. 基于关联主义的大规模网络开放课程(MOOC)及其学习支持[J]. 远程教育杂志, 2012, 30(3): 31-36

[24] Wing J M. Computational thinking[J]. Communications of the ACM, 2006, 49(3): 33-35

[25] Byrne M D, Catrambone R, Stasko J T. Do Algorithm Animations Aid Learning?: GIT-GVU-96-18[R]. Georgia Institute of Technology, 1996

[26] Byrne M, Catrambone R, Stasko J. Evaluating animations as student aids in learning computer algorithms[J]. Comput. & Educ., 1999, 33(4): 253-278

[27] Kehoe C, Stasko J, Taylor A. Rethinking the evaluation of algorithm animations as learning aids: An observational study[J]. Hum-Comput Studies, 2001, 54(2): 265-284

[28] Vector B. Inventing on principle[OL]. https://vimeo.com/36579366