一种时间依赖路网最小时间路径规划算法研究*

孙 奥[1], 朱桂斌[1], 江 铁[1], 史名一[2]
(1. 重庆通信学院 应急通信重庆市重点实验室,重庆 400035; 2. 71521 部队, 河南 驻马店 463000)

摘 要: 研究时间依赖路网(TDN)的最短路径规划算法,对指导人们出行和解决城市交通等问题具有十分重要的意义。在研究前人算法的基础上,提出了一种利用结构体数组来求解 TDN 路网最小时间路径规划算法。对算法的基本原理和结构体数组的构造进行了介绍,对算法实现流程及其中一些关键步骤进行了重点阐述,最后在 VC ++ 环境中利用 MapX 控件对算法进行了实验仿真。仿真结果表明,该算法具有较高的搜索效率,且能适应路况变化,基本满足现实需要。

关键词: 路径规划; 最小时间; 时间依赖路网; 行程时间; 结构体数组
中图分类号: TP301.6 **文献标志码**: A **文章编号**: 1001-3695(2012)11-4148-04
doi:10.3969/j.issn.1001-3695.2012.11.038

Time-dependent road network and minimum time path planning algorithm

SUN Ao[1], ZHU Gui-bin[1], JIANG Tie[1], SHI Ming-yi[2]
(1. *Chongqing Key Laboratory of Emergency Communication*, *Chongqing Communication Institute*, *Chongqing* 400035, *China*; 2. *Troops* 71521, *Zhumadian Henan* 463000, *China*)

Abstract: It has great significance of the algorithm about the shortest path planning based on the time-dependent road network (TDN). Because it can guide people to travel, solve the urban traffic and other issues. This paper proposed a structure array TDN road network to solve the minimum time path planning algorithm on the basis of previous algorithm. Firstly, it introduced the basic principle of the algorithm and the structure of the array. Then it focused on the implementation and some of the key steps of the algorithm. At last, it used MapX to test based on VC ++ environment. The experiment results show that it has high search efficiency. It can adapt to the traffic changes and basically meet the needs of people.

Key words: path planning; minimal time; time-dependent road network; travel time; structure array

0 引言

由城市道路组成的交通路网是一个内部关系复杂的时间变化网络,因其时间特性而称之为时间依赖网络(简称 TDN)[1]。研究 TDN 的路径规划问题对指导人们出行并解决当今日益突出的城市交通等问题都具有十分重要的意义。

TDN 路径规划算法已成为动态路径规划技术的研究热点,不少专家学者从不同的角度展开了研究。Dreyfus[2] 最早提出改进原有的经典算法如 Dijkstra 算法可以求解 TDN 网络的最小时间路径问题,但随后 Kaufman 等人[3] 给出了传统搜索算法不能满足 TDN 网络的反例,并提出建立严格的一致性条件来限制"病态"实例的出现。谭国真等人[4] 从理论上证明了经典规划算法如 Dijkstra 算法不能有效求解 TDN 网络最短路径的问题,并给出了 SPTDN 算法。何俊等人[5] 又重新证明了谭国真等文章中相关定理,同时也给出了一种适用的最短路径算法。李星毅等人[6] 提出了一种改进算法通过引入一个记录路径的数组不仅有效解决了文献[4]不能正确求解最优路径的问题,而且还扩展了原算法的应用范围。余伟辉等人[7] 基于扩散法思想,提出了一种求解时间依赖有向无环网络中的最小时间路径规划改进算法,有效地解决了扩散算法不能确定停止时间和算法复用问题。章昭辉[8] 提出了一种基于贪心策略的离散变权网络中的次优路径规划算法,大大提高了搜索效率。本文在研究相关文献的基础上提出了一种利用结构体数组来求解 TDN 时间最小路径问题。

1 算法思想及流程

1.1 算法思想

文献[4~6]中介绍的方法都以维护一个队列表来完成节点的搜索,并通过一个数组来记录路径。基于该思想,本文算法中用到的结构体数组也应至少具备以上功能,一种简单的结构体数组设计如下定义代码所示。

```
typedef struct OPENTABLE//遍历节点表
{
  long Start;//起点 ID
  long End;//终点 ID
```

收稿日期: 2012-04-04; **修回日期**: 2012-05-08 **基金项目**: 重庆市科技攻关项目(CSTC,2010AC2037)

作者简介:孙奥(1983-),男,湖北当阳人,硕士研究生,主要研究方向为实时路况动态车载导航关键技术(ziao212@163.com);朱桂斌(1972-),男,副教授,硕导,博士,主要研究方向为信息处理、模式识别;江铁(1987-),男,四川绵阳人,硕士研究生,主要研究方向为数字图像融合处理;史名一(1983-),男,河南驻马店人,参谋,主要研究方向为通信工程和用频筹划.

```
    long line;//路段 ID
    int isSun;//表示当前节点是否被遍历过,1 表示是,0 表示否
    float AllTime;//遍历后到当前点的时间总花费
} OT;
OT ot1[POINTNUM];
```

为更好地阐述本算法的基本思想,结合节点搜索示意图来说明结构体各变量在节点搜索过程中的作用。节点搜索示意图如图 1 所示。

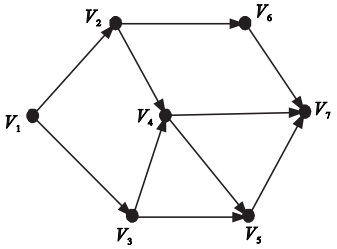

图1　节点搜索示意图

节点在搜索过程中,搜索到的每个节点 V_i 都将以一个结构体数组元素存储在结构体数组 ot1[j]中(i 和 j 不一定相等),以当前节点为起点的路段有 N 条,则将在数组 ot1[ ]中相应地存储 N 个节点结构体数组元素,但是以当前点为终点的路段有 M 条,仅在数组 ot1[ ]保存到达该节点时间最小的那条路段信息,搜索过程中,所有数组元素都将以变量 end 为基准来确定其他字段信息。如图 1 所示,以节点 V_1 出发的路段有 V_1V_2 和 V_1V_3 两条,则在数组中增加节点 V_2 和 V_3 的数组元素信息记录(若数组中没有其记录);以节点 V_4 为终点的路段也有两条,分别是经由节点 V_2 和 V_3,若节点 V_1 经节点 V_2 到达节点 V_4 的时间比经由节点 V_3 到达节点 V_4 的时间短,则在数组表保存的节点 V_4 的相关信息中,其 start 节点就是节点 V_2,line 编号即为 V_2V_3 路段 ID, AllTime 值是从起点 V_1 经由 V_2 到达 V_4 的时间值。结构体中字段 isSun 用以表示当前节点的搜索状态,当前节点若已有后续节点,则记为 0,若没有则记为 1。当节点搜索到目标点时,若结构体数组中所存储的节点信息中变量 isSun 为 1 的所有节点数组元素的 AllTime 数值都比目标点的 AllTime 值大,则当前记录的目标点信息即为最后结果。

最后时间最小路径由目标节点变量 start 和 line 信息向起点方向反推即可得到所求的时间最短路径(当前节点的变量 start 是上一个节点的变量 end)。

需要强调的是,在搜索节点过程中,若结构体数组表中已经存储到了该节点的相关信息,若新得到的 AllTime 值要小则更新该节点的相关信息,若新得到的 AllTime 值要大,则不用更新,但是记录当前节点的变量 isSun 为 0;若结构体数组表中没有该点的相关信息,则在数据表中增加其节点信息,并将其变量 isSun 记为 1。

1.2　算法流程

总结算法的执行过程,其算法流程如图 2 所示。

程序开始要分别对路径规划所涉及的路段图层和节点图层以及对应的数据库数据进行初始化,数组实例的初始化时每个结构体的节点编号设为 0,AllTime 设为 0,isSun = 0。以上流程中没有考虑由于地图区域限制找不到路径的情况,若要考虑,在流程中增加一个退出判断即可。

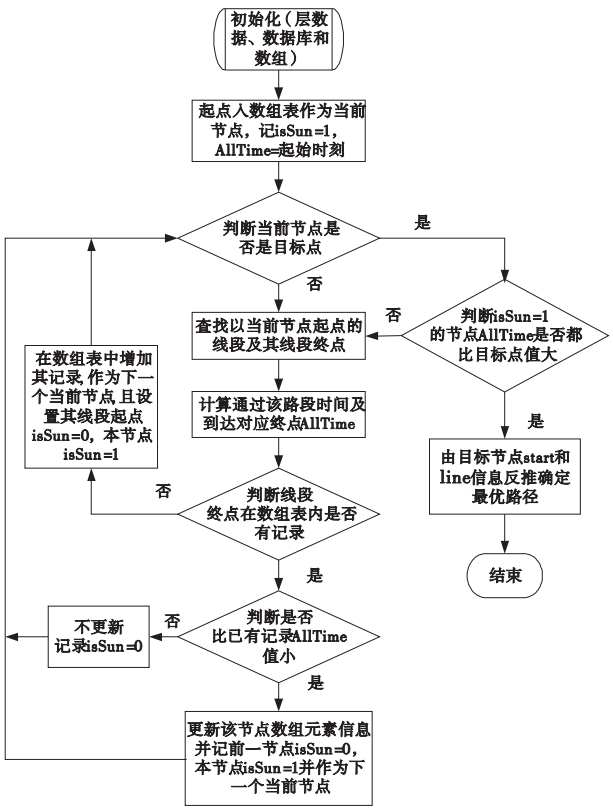

图2　结构体数组规划算法节点搜索流程

2　关键步骤

2.1　路网离散时间模型

按文献[6]中介绍的时间离散模型思想对 TDN 网络进行离散化,将时间区间$[t_0,t_d]$(t_0 为起点时刻,t_d 为到达目标点的预计时刻)分为$[t_0,t_0+\Delta t]$,$[t_0+\Delta t,t_0+2\Delta t]$,…,$[t_0+(N-1)\Delta t,t_0+N\Delta t]$,$[t_0+N\Delta t,t_d]$,将短时间段 Δt(与交通网络短时预测时间相同,一般为 5 min)内的交通状况看做是近似不变的,即每一条道路上的平均行驶速度为一恒定值,且可由相应的预测模型进行确定,可记做离散时间依赖网络(V,E,D)。其中,V 为有限节点集;E 是有限弧集(道路集);$E=V\times V$,T_{ij}是每一条道路$(V_i,V_j)\in E$ 在时间闭区间上定义的时间代价函数(本文指路段行程时间)。

为了更好地说明本路径规划算法搜索过程,本文将路况等条件进行了必要的优化:a)所有的道路均是双向的,不考虑限行及其他交通规则;b)在路网中行驶不停靠,不掉头,不考虑交通事故、堵车等情况;c)不考虑天气等情况的影响;d)假定每条道路上最高行驶速度 v_{max}是一个定值。

2.2　某路段的行程时间计算

实际的路段行程时间计算比较复杂,有不少文献进行了专门研究,一般将其分为路段自由行驶时间和通过交叉路口延误时间[9]等,并利用经典的道路行程时间计算公式 BPR 等函数计算得到。由于路段行程时间不是本文的研究重点,本文仅为说明问题,特将行程时间计算问题简单化,不考虑路段的复杂情况,仅以路段某时段内的平均行驶速度(假设可由相关预测技术精确预测)来间接计算行程时间。某条路径上的行驶时间根据车辆的行驶状态可分为以下几种情况:

a)时间段 Δt 内直接通过路段(V_i,V_j):

$$T_{ij}=\frac{l_{ij}}{v_{ij}(n,\Delta t)}$$

b)时间段 Δt 内不能通过路段(V_i,V_j):

$$T_{ij}=k\Delta t+\frac{(l_{ij}-\Delta t\times\sum_{n}^{n+k-1}v_{ij}(n,\Delta t))}{v_{ij}((n+k),\Delta t)}\text{或}$$

$$T_{ij}=\Delta t-\overline{T_{ui}}+\frac{l_{ij}-(\Delta t-\overline{D_{ui}(t)})\times v_{ij}(n,\Delta t)-\Delta t\times\sum_{n+1}^{n+k}v_{ij}((n+1),\Delta t)}{v_{ij}((n+1+k),\Delta t)}+$$

$$k\Delta t=(k+1)\Delta t-\overline{T_{ui}}+$$

$$\frac{l_{ij}-(\Delta t-\overline{T_{ui}})\times v_{ij}(n,\Delta t)-\Delta t\times\sum_{n+1}^{n+k}v_{ij}((n+1),\Delta t)}{v_{ij}((n+1+k),\Delta t)}$$

其中:T_{ij}表示路段(V_i,V_j)上的行驶时间;l_{ij}表示路段(V_i,V_j)的长度;$v_{ij}(n,\Delta t)$表示第 n 个时间段 Δt 内路段(V_i,V_j)的平均速度;k 表示在路段(V_i,V_j)上经过时间段 Δt 的数目($k\geqslant 1$),$\overline{T}_{ui}=\sum T_i-t_0-(n-1)\Delta t$,$\overline{T}_{ui}$表示第 n 个时间段 Δt 内在路段(V_u,V_i)上行驶时间,t_0 为起点时刻,$\sum T_i$ 表示行驶到节点 i 时的时刻;$\Delta t-\overline{D_{ui}(t)}$表示在路段$(V_u,V_i)$上最后一个时间段 Δt 内行驶完该路段后的剩余时间;(V_u,V_i)为(V_i,V_j)的前相邻路段。

2.3 路段速度的表示、存储及读取

在根据某一地区导航电子地图进行路径规化时,由于所用导航电子地图无论是节点(路口)还是路段的编号都不一定是从 1 开始编号,而且也不一定是顺序编号。一般情况下,一个城市的导航电子地图数据十分庞大,加之编号规则,节点和路段的编号可能是一个很大的数字。因此,对某一地区的道路进行路径规划时,如果直接开始从起始编号 0 进行搜索并保存无疑会占用较多内存和时间。而且在外接数据库的路段速度表中,每个路段都有多个时段的预测速度,在读取某路段某时段内的速度数值时若每次遍历路段编号无疑也会增加搜索时间。本文算法在实现过程中提出了一个解决办法:

a)定义一个路段的结构体数组 line l[LINE_NUM]。其变量结构如下:

```
typedef struct stcLine
{
   long start;//起点
   long end;//终点
   long line;//路段编号
   long length;//长度
} Line;
Line l[LINE_NUM];
```

b)定义一个与路段数组编号相匹配的路段速度数组 ls[LINE_NUM][TIME_NUM]。

c)路段速度数据表按一定顺序排列,如 line_speed order by line_id,time_id。

路段数组的增加可以在初始化时对该区域地图内的路段重新编号,虽然在初始化时会增加一定的初始化时间,但是却大大减小了程序对路段编号遍历的时间和因路段原始编号而浪费的内存。数组 ls[LINE_NUM][TIME_NUM]由于与路段数组编号相匹配,在计算路段时间时,依次读取路段速度值可减少对路段编号的遍历,节省了遍历时间。这一过程是在数据库按一路段编号、时间编号顺序排列的基础之上的。

3 实例仿真

3.1 条件设置

仿真实验所用导航电子地图为重庆市江北观音桥地区简易电子地图。地图含路段数量 274 条,节点(路口)数量 233 个,实验仿真用电脑处理器配置为 Intel® Core™2 Duo CPU T6600 2.20 GHz,内存 1.99 GB,在 VC ++6.0 环境下,利用 MapX 的二次开发相关技术完成。由于所用地图区域原因,为更好地验证算法的正确性及适用性,路况信息更新周期设为 2 min。所用路段行驶速度信息表截图如图 3 所示(line_id 为路段编号,time_id 是根据离散模型思想划分的时间段,speed 为路段在不同时段内的行驶速度预测值)。

line_id	speed	time_id
20009	22	1
20009	18	2
20009	32	3
20009	25	4
20009	28	5
20009	26	6
20009	23	7
20009	12	8
20009	7	9
20009	43	10
20010	33	1
20010	36	2

图3 路段行驶速度信息表截图

3.2 相同起止点不同路况下路径规划仿真实验

实验中,用三张速度表分别表示不同的路况信息。通常情况下,路况不同,在相同起止点规划出的路径应有所不同。设置起点为新村小学(ID22001),终点为妇幼保健院(ID31002),算法根据不同的速度表规划出的最小时间路径如图 4 ~ 6 所示。

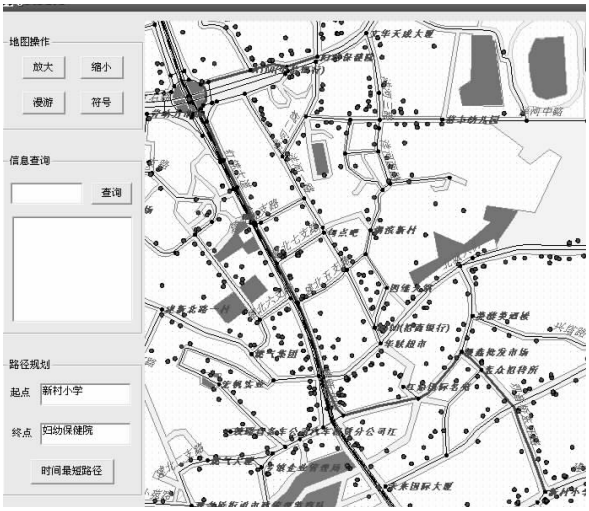

图4 速度表1下路径规划结果

这三种路况下的测试数据如表 1 所示。

以上实验仿真数据表明,此算法能够实现 TDN 路网的最小时间路径规划,且对不同的路况具有较好的适应能力,能根据路况变化及时更新路径规划结果,且对地图节点的搜索速度也比较快,具体搜索时间会因规划结果稍有变化。

图5　速度表2下路径规划结果

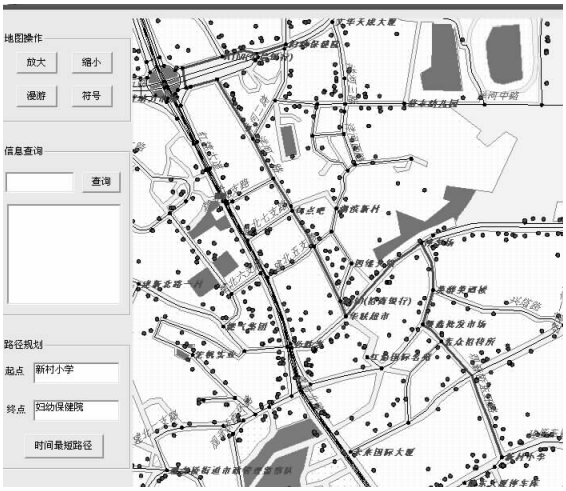

图6　速度表3下路径规划结果

表 1　不同路况下的路径规划测试结果

路况	起点	终点	路径长度/m	行驶时间/min	搜索时间/ms
速度表 1	22001	31002	2093	4.240541	78
速度表 2	22001	31002	2089	6.266999	78
速度表 3	22001	31002	2186	3.739247	63

4　结束语

本文通过对时间 TDN 路径规划问题进行研究,提出了一种以结构体数组来完成节点搜索,实现最小时间路径规划算法。通过仿真实验验证了该算法同样具有较好的性能,能较好地满足 TDN 路网最小时间路径规划要求。但是该算法对路网模型考虑比较简单,如没有考虑交通规则、交通堵塞等复杂情况,具体的应用还有待实际检验。对算法进行改进或优化(如设置启发信息增强搜索方向性、结构体增加变量体现交通规则等复杂情况等)有待下一步作深入研究。

参考文献:

[1] 谭国真. 时变、随机网络最优路径算法及其应用研究[D]. 大连:大连理工大学,2002.

[2] DREYFUS S E. An appraisal of some shortest path algorithms[J]. Operations Research,1969,17(3):395-412.

[3] KAUFMAN D E, SMITH R L. Fastest path in time-dependent network for intelligent vehicle-highway systems application[J]. IVHS Journal,1993,11(1):1-11.

[4] 谭国真,高文. 时间依赖的网络中最小时间路径算法[J]. 计算机学报,2002,25(2):1-8.

[5] 何俊,戴浩,宋自林,等. 时间依赖的交通网络模型及最短路径算法[J]. 解放军理工大学学报:自然科学版,2005,6(6):541-544.

[6] 李星毅,翟晓峰,施化吉. 最小时间路径算法的改进及在路径优化中的应用[J]. 计算机应用研究,2008,25(6):1645-1647.

[7] 余伟辉,陈闵中. 时间依赖有向无环网最小时间路径算法[J]. 计算机工程与科学,2008,30(11):42-45.

[8] 章昭辉. 一种基于离散变权网络的动态最短路径快速算法[J]. 计算机科学,2010,37(4):238-240.

[9] 张廷. 城市道路行程时间预测研究[D]. 长沙:湖南大学,2010.

(上接第 4116 页)的隐层神经元至输出神经元的权值(即未添加该神经元时的权值),只确定新添加的隐层神经元的权值,而本文则是延续了笔者团队之前研究的基于伪逆的方法,这样便可在增加隐层神经元时直接一步计算出连带之前隐层神经元的全部连接权值[10]。总的来说,上述两种方法在确定隐层神经元数目和权值时有着明显的不同,这从某种程度上也彰显了本文算法的创新之处。

4　结束语

本文展示了一种多输入单极性 Sigmoid 激励函数神经网络模型,同时为了克服传统 BP 神经网络权值和结构确定过程中的固有缺陷,采用了权值直接确定法来快速确定网络最优权值,进而提出了双阶段结构自确定法,该算法能够快速有效地确定神经网络最优结构。计算机数值实验结果进一步表明,基于该双阶段结构自确定法的神经网络具有良好的学习和泛化能力,在多输入函数逼近方面体现出较优良的性能。

参考文献:

[1] BISHOP C M. Neural networks for pattern recognition[M]. New York: Oxford University Press,1995.

[2] 张青贵. 人工神经网络导论[M]. 北京:高等教育出版社,1992.

[3] 成素梅,郝中华. BP 神经网络的哲学思考[J]. 科学技术与辩证法,2008,25(4):20-25.

[4] 张祥敢,刘长安,方文涛. 基于改进 BP 神经网络的控制图模式识别系统[J]. 组合机床与自动化加工技术,2011(9):43-46.

[5] 孙彬,李铁克,张文学. 基于结构修建神经网络的股票指数预测模型[J]. 计算机应用研究,2011,28(8):2840-2843.

[6] 周政. BP 神经网络的发展现状综述[J]. 山西电子技术,2008(2):90-92.

[7] 苏高利,邓芳萍. 论基于 MATLAB 语言的 BP 神经网络的改进算法[J]. 科技通报,2003,19(2):130-135.

[8] HUANG Guang-bing, ZHU Yu-qin, SIEW C K. Extreme learning machine: theory and applications[J]. Neurocomputing, 2006, 70(1-3):489-501.

[9] 张雨浓,李凌峰,郭东生,等. Hermite 插值神经网络权值和结构确定理论探讨[J]. 计算机应用研究,2010,27(11):4048-4051.

[10] 张雨浓,杨逸文,李巍. 神经网络网络权值直接确定法[M]. 广州:中山大学出版社,2010.

[11] YU Hao, WILAMOWSK B M. Levenberg-Marquardt training[J]. Intelligent Systems,2010(12):1-6.

[12] HUANG Guang-bing, CHEN Lei, SIEW C K. Universal approximation using incremental constructive feedforward networks with random hidden nodes[J]. IEEE Trans on Neural Networks,2006,17(4):879-883.

学习样本集$\{(\boldsymbol{X}_n=[x_{n,0},x_{n,1},x_{n,2}]^{\mathrm{T}},\delta_n)\}_{n=0}^{N-1}(N=343)$。使用上述双阶段结构自确定法对网络进行训练,校验区间为$[0.06,0.925]^3$,以0.15的间隔采集样本点进行校验,结果如表2所示。

表2 神经网络对三输入目标函数6~10的学习与校验结果

目标函数	算法执行时间/s	最优隐层神经元数/个	学习误差	校验误差
6	7.637	127	1.702×10^{-10}	1.608×10^{-10}
7	7.707	127	3.924×10^{-10}	4.132×10^{-10}
8	7.384	126	2.069×10^{-10}	1.752×10^{-10}
9	5.704	103	4.906×10^{-12}	3.656×10^{-12}
10	12.172	151	1.295×10^{-9}	4.714×10^{-9}

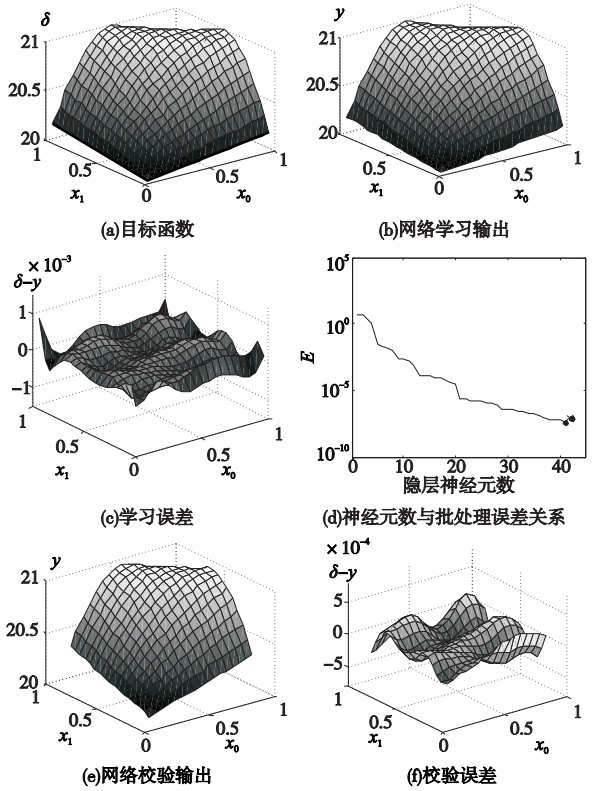

图5 二输入目标函数3的神经网络数值实验结果

3.3 四输入函数学习与校验

在进行四输入数值实验时,学习样本区间为$[0.3,0.8]^4$,以采样点间隔0.08进行均匀采样,从而得到学习样本集$\{(\boldsymbol{X}_n=[x_{n,0},x_{n,1},x_{n,2},x_{n,3}]^{\mathrm{T}},\delta_n)\}_{n=0}^{N-1}(N=2401)$,并使用上述算法对网络进行训练。然后在$[0.32,0.79]^4$内沿着$x_0$、$x_1$、$x_2$和$x_3$方向以0.09的间隔采集样本点进行校验,得到如表3所示结果。以第13个目标函数为例,双阶段结构自确定法最优结构搜索过程和校验样本误差如图6所示。

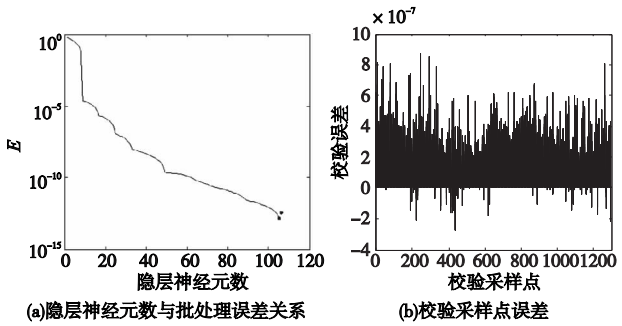

图6 四输入目标函数13的神经网络数值实验结果

表3 神经网络对四输入目标函数11~15的学习与校验结果

目标函数	算法执行时间/s	最优隐层神经元数/个	学习误差	校验误差
11	70.86	169	5.401×10^{-11}	3.803×10^{-11}
12	52.639	137	7.767×10^{-13}	7.503×10^{-13}
13	42.475	106	6.360×10^{-14}	8.892×10^{-14}
14	125.521	209	6.009×10^{-15}	4.796×10^{-15}
15	87.921	193	4.135×10^{-11}	2.673×10^{-11}

为了更进一步展现本方法的优势,本文同时采用了改进的BP迭代法,即Levenberg-Marquardt(LM)算法[11],进行神经网络逼近多输入目标函数的计算机实验验证。现选取四输入目标函数11~15用LM算法训练单隐层Sigmoid激励函数的神经网络,展示它们对目标函数的学习和校验效果,实验结果如表4所示。另外,在迭代过程中,设定迭代最长时间为600 s。

表4 LM算法对四输入目标函数11~15的学习与校验结果

目标函数	算法执行时间/s	隐层神经元数/个	学习误差	校验误差
11	600	169	7.345×10^{-9}	1.331×10^{-8}
12	600	137	2.297×10^{-9}	8.695×10^{-8}
13	600	106	3.028×10^{-9}	2.606×10^{-9}
14	600	209	3.695×10^{-9}	4.209×10^{-6}
15	600	193	1.268×10^{-8}	2.128×10^{-8}

通过表4与3的对比可以看出,基于LM算法的神经网络虽然已达到了设定的最大迭代时间600 s,但是其学习误差和校验误差较本文提出的方法所对应的误差要大很多(相差了几个数量级)。值得指出的是,运用LM算法构造的神经网络,需要预先设定隐层神经元数目(这里采用的是表3中所确定的隐层最优神经元数目);同时,如果要更好地完成对网络的训练,还需要尝试多次不同的隐层神经元数目。相比之下,本文提出的结构确定方法可以有效地解决隐层神经元数难以确定这个问题。

从上述的二、三和四输入目标函数逼近结果以及与LM算法的对比结果可以看出,本文提出的基于权值直接确定法的双阶段结构自确定法是可行且有效的。同时,表1~3的数据也显示了通过本算法所得到的最优神经网络学习误差小,数量级在10^{-14}~10^{-8}左右;网络校验误差小,数量级在10^{-15}~10^{-8}左右,即无过拟合现象发生;另外,算法运行时间也较短,证明了该算法的高效性。实际上,笔者也针对其他更多的输入的情况进行了相应的数值实验,对应的实验结果(限于篇幅就没有展示出来)也显示出本文所提出的基于权值直接确定法的双阶段结构自确定法的有效性和可行性。

值得指出的是,本文是基于笔者和其团队之前工作的进一步探索所取得的成果。虽说本文所提出的方法与增量ELM方法[12]之间在某些方面有一定的关联,但还是有很大的不同,列举如下:a)增量ELM方法采取了逐个增加隐层神经元的方法,而本文则提出了双阶段结构确定方法(图3和4)来确定隐层神经元数目;b)增量ELM方法采取了人工设定期望精度及最大隐层神经元数目的方式,并以此作为该结构确定算法的停止点[12],相比之下,本文提出的双阶段结构确定方法无须上述人工干预,只需给出相应的学习样本,便可自动计算出最佳逼近误差以及确定最优网络结构;c)增量ELM方法在每增加一个新的隐层神经元时,不重新计算之前已确定 (下转第4151页)