算法第六次作业

2020年6月21日 21:48

解装载问题的分支限界算法中，由EnQueue产生的结点可以在算法结束前一次性删除。然而那些没有活儿子结点或没有叶结点的扩展结点可以立即被删除。试设计一个在算法中及时删除不用结点的方案，并讨论其时间与空间之间的折中。

解：思路是确定一个是否删除的变量，在遇到左右儿子都被剪枝的时候删除该结点。

代码：

```
template<class Type>
Type MaxLoading(Type w[], Type c, int n, int bestx[])
{
	//队列式分支限界法，返回最优载重量，bestx返回最优解
	//初始化
	Queue<QNode<Type> *> Q;		//活结点队列
	Q.Add(0);				//同层结点尾部标志
	int i = 1;				//当前扩展结点所处的层
	Type Ew = 0,				//扩展结点所相应的载重量
		bestw = 0,			//当前最优载重量
		r = 0;				//剩余集装箱重量
	for (int j = 2; j <= n; j++)
		r += w[j];
	QNode<Type> *E = 0,			//当前扩展结点
		*bestE;				//当前最优扩展结点
	while (true)
	{					//搜索子集空间树

		bool flag = false;		//判断结点是否删除标志

		//检查左儿子结点
		Type wt = Ew + w[i];
		if (wt <= c)
		{				//可行结点
			if (wt > bestw)
				bestw = wt;
			EnQueue(Q, wt, i, n, bestw, E, bestE, bestx, true);
			flag = true;
		}
		//检查右儿子结点
		if (Ew + r > bestw)
		{
			EnQueue(Q, Ew, i, n, bestw, E, bestE, bestx, false);
			flag = true;
		}
		if(E)				//结点删除标志未改变，左右儿子均被剪枝，删除该结点
			if(!flag)
				delete E;

		Q.Delete(E);			//取下一层扩展结点
		if (!E)
		{				//同层结点尾部
			if (Q.IsEmpty())
				break;
			Q.Add(0);		//同层结点尾部标志
			Q.Delete(E);		//取下一层扩展结点
			i++;			//进入下一层
			r -= w[i];		//剩余集装箱重量
		}
		Ew = E->weight;			//新扩展结点所相应的载重量
	}
	//构造当前最优解
	for(int j=n-1;j>0;j++)
	{
		bestx[j] = bestE->LChild;
		bestE = bestE->parent;
	}
	return bestw;
}
```