

根据序列变化率预测软件阶段成本

王 勇 宋擒豹 沈钧毅

(西安交通大学计算机科学与技术系 西安 710049)

摘 要 针对软件阶段成本因少数据、不确定性使得用现有方法(如回归)难以预测的问题,文中提出一种新颖的预测方法,该方法从项目已完成阶段的成本序列中,通过变换得到反映序列变化快慢的“变化率”,并用机器学习方法从历史项目中学习得到变化率阈值,然后用不同的灰色模型进行预测.在10个现实世界软件工程数据集上的实验结果表明,该方法平均预测误差比线性回归方法低20%~80%,显示出较大的潜力.

关键词 项目成本预测;软件成本;阶段成本;灰色模型

中图法分类号 TP311

DOI号: 10.3724/SP.J.1016.2009.01346

Predicting Software Stage Effort with Sequence Changing Ratio

WANG Yong SONG Qin-Bao SHEN Jun-Yi

(Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049)

Abstract Software stage effort has the features of data starvation and uncertainty. It is difficult to use the current methods (e. g. regression) to make predictions. This paper proposes a novel prediction method, which gets the effort sequence changing feature — “changing ratio” from the completed stage effort sequences, and gets the “changing ratio threshold” from historical projects by machine learning methods, then uses grey models to make predictions. The experimental results on 10 real world software engineering datasets show that, compared with linear regression method, the prediction accuracy of the proposed method has been improved by 20%~80%. This is very encouraging and indicates that the method has considerable potential.

Keywords project cost prediction; software cost; stage cost; grey model

1 引 言

软件成本预测是项目管理的重要组成部分,也是软件项目管理领域最具挑战性的任务之一^[1]. 缺少有效的软件项目成本预测而导致项目超期或失去控制一直是一个突出问题^[2-5]. 除了需要在项目早期预测总体成本,以便做出合理的投标决策及项目计划外^[2,6-15],在项目开发过程中动态地预测各阶段的成本也尤为重要^[16]. 这种动态预测可使管理人员发

现可能出现的成本进度方面的问题,提前采取有效措施,保障项目顺利进行.

软件开发是一种创造性劳动,有很多无法确知的因素影响开发成本,如开发人员的技能、项目的复杂程度等. 这些因素导致软件成本存在很大的不确定性,难以准确预测^[17]. 然而,所有这些因素的综合作用都会随着项目的进展在成本上反映出来,因此可以通过研究成本前期的变化,发现其演变规律,进而对后续成本进行预测. 同时,软件项目管理过程中产生的数据量有限,因此预测还必须在少数据、不确

定的环境下进行. 传统的基于机器学习和统计的方法(如回归)因为需要大量的数据,而且要求数据满足特定的分布^[18-19],不适合这种预测,因此,需要探索新的方法.

在控制论中,内部信息缺乏的对象和系统称为黑箱,黑表示信息缺乏,白表示信息完全,灰介于黑和白之间,表示信息贫乏或不确定^[20]. 邓聚龙教授于 1982 年首次提出灰色系统理论^[21],以“小样本”、“贫信息”不确定性系统为研究对象,通过对已知信息的生成、提炼,揭示系统内在发展规律,实现对系统行为的描述、预测和控制^[22]. 该理论一个突出的优点是只需少量实验观测数据就可以建立用于预测的灰色模型,不要求数据符合特定的概率分布^[22-23],已广泛应用在能源^[24]、通信^[25]、测量^[26]、图像处理^[27]等许多领域,取得了很好的效果. 同时,宋擒豹和 Shepperd 等^[28]已将灰色系统理论中的灰关联分析用于软件过程早期的总体成本预测并取得成功. 本文探索用灰色系统理论中的灰色模型预测软件项目阶段成本的问题.

本文提出一种根据软件阶段成本序列及其变化率构造灰色模型进行软件阶段成本预测的方法 GMCP(Grey Models for software Cost Prediction). 该方法首先根据项目已完成阶段的成本数据及其变化率,动态构造 GM(1,1)或 Verhulst 模型,模拟出成本序列的发展趋势,进而预测后续阶段的成本,并采用误差补偿技术来提高预测精度.

本文其余部分的结构如下,下一节介绍软件阶段成本预测的研究现状,然后介绍灰色系统理论的基本概念和灰色模型的预测原理,接下来详细讨论 GMCP 的实现过程,最后分析实验数据和结果,并给出结论.

2 相关工作

目前,软件成本预测的研究主要集中在项目早期对总成本的预测上^[2,6-15],对软件阶段成本预测的研究还非常少. 目前,在世界范围内,已有的研究工作主要有 3 项,分别是 MacDonell 和 Shepperd^[29]利用线性回归模型预测软件生命周期阶段成本; Kulkarni 等^[30]用迷你模型(mini-model)方法预测阶段成本以及 Ohlsson 和 Wohlin^[31]根据阶段结束后获得的新信息改进先前的预测. 用灰色系统理论预测软件总体成本方面宋擒豹和 Shepperd 等^[28]做了相关的研究工作.

关于阶段成本预测的 3 项研究都是针对软件生命周期阶段,如计划、设计、测试等. MacDonell 等人通过建立线性回归模型用项目前期阶段成本预测后续阶段的成本. 在 16 个类似的软件项目上,对任意两个阶段成本做相关性检验,选择相关程度较大的阶段建立回归模型. 例如,对计划和测试建立回归模型,根据前者预测后者. 结论指出,这种方法和专家方法^[2,7]结合可以改进预测精度. Ohlsson 等人^[31]的研究基于 26 个相似的项目,主要思想是用前阶段结束后获得的新信息改进原有的预测. 例如,需求阶段完成后,用需求规格书中需求的个数作为输入,通过预测模型得到新的工作量预测. 但是结果表明这个方法并没有提高预测精度,因为模型的输入量和工作量的相关性不显著. 同一种输入,对一些项目的工作量相关性很强,对另一些项目很弱或相关性相反. 因此,确定合适的输入是该模型的难点. Kulkarni 等人^[30]用转换矩阵建立阶段输入输出间的预测模型. 他们为每个阶段建立一个转换矩阵,称作“迷你模型”(mini-model),这个矩阵可以把本阶段的输入转换成输出. 例如,某阶段的输入是数据流图的个数,经过矩阵转换,输出得到子功能个数、代码行数等,同时作为下个阶段的输入. 把矩阵串联起来,提供第一个阶段的输入,可以得到任意阶段的输出. 如果前一个矩阵的输出和后一个所需的输入不匹配,则需要进行剪裁. 这种方法应用于一个军方的项目,实验的数据只是用于演示,并未给出预测精度.

3 种方法使用的输入也各不相同. MacDonell 等人^[29]用人小时这样的经验(empirical)数据, Ohlsson 等人^[31]用需求个数、测试用例个数、过程个数等基于阶段文档的对象, Kulkarni 等人^[30]用传统的代码行、功能点数、程序包个数等. 文献[30-31]中所用的代码行数、测试用例个数等只适用于特定类型的项目,当项目类型变化时需重新选择度量方法,而选取合适的度量是这类方法最关键和困难的部分. 本文采用目前使用最广泛的人小时工作量数据作为输入,和文献[29]中使用的相同.

MacDonell 等人在文献[29]中考察项目阶段两两间的相关性,根据这种相关性来做预测. 事实上,阶段间并不都存在明显的相关性,因此不是所有的阶段都能用这个方法预测,文中只预测了设计、实现、测试 3 个阶段. 本文的方法不要求阶段成本间存在相关性,适用于除最初用于建模的几个阶段外的所有阶段.

宋擒豹和 Shepperd 等^[28]用灰色理论对软件总

体成本预测方法进行了研究. 他们使用灰色关联分析方法首先选取影响成本较大的特征因子, 然后通过这些因子找出和被预测项目最相关的项目进行预测, 取得了很好的效果. 虽然都使用灰色理论, 本文使用的方法及目标和文献[28]有很大差别. 首先, 本文致力于预测软件阶段成本而非总体成本; 其次, 使用的灰色模型方法和文献[28]中的灰色关联分析是两种原理、用途都差异很大的方法.

从上面的介绍可以看出, 关于软件阶段成本预测的研究很少, 还没有使用灰色理论进行此类预测的研究. 本文提出的基于灰色模型的预测方法, 和前面的方法有显著的不同, 可作为一种可选方法.

3 基于灰色模型的软件阶段成本预测

为了预测软件阶段成本, 我们从软件成本进度曲线(rayleigh curve)^[32]中得到启发, 即软件开发过程中所需的人员数量和进度之间存在函数关系, 可用进度曲线来表示. 那么, 如果能够找到阶段成本和进度间的某种函数关系, 就可以利用它来预测阶段成本. 然而, 原始的阶段成本数据过于离散, 没有明显的规律, 难以用一个函数或曲线来模拟. 但是, 在对序列做累加生成后, 会显示出较明显的准指数规律, 可以找到一条指数曲线对累加序列进行拟合, 进而预测未来的发展, 这条曲线可以用灰色模型来表示. 根据拟合的不同方法, 灰色模型有多种类型, 主要包括 GM(1, 1)、Verhulst、DGM 等. 本文使用 GM(1, 1)和 Verhulst 相结合进行预测. 下面介绍 GM(1, 1)、Verhulst 的预测过程^[22]以及使用灰色模型进行软件阶段成本预测的方法.

3.1 GM(1, 1)模型的预测过程

GM(1, 1)的含义是一个变量一阶微分方程, 主要用来模拟单调的准指数序列, 其预测步骤如下.

设 $X^{(0)} = (x^{(0)}(1), x^{(0)}(2), \dots, x^{(0)}(n))$ 是非负原始序列.

第 1 步. 累加生成. 对 $X^{(0)}$ 做一阶累加生成, 得到数列

$$X^{(1)} = (x^{(1)}(1), x^{(1)}(2), \dots, x^{(1)}(n)) \tag{1}$$

其中,

$$x^{(1)}(k) = \sum_{i=1}^k x^{(0)}(i), \quad k=1, 2, \dots, n \tag{2}$$

第 2 步. 用灰指数曲线拟合 $X^{(1)}$. 假设曲线满足下面的方程:

$$\frac{dx^{(1)}}{dt} + ax^{(1)} = b \tag{3}$$

方程的解为

$$\hat{x}^{(1)}(t) = \left(x^{(0)}(1) - \frac{b}{a}\right)e^{-at} + \frac{b}{a} \tag{4}$$

因为 $X^{(1)}$ 是离散序列, 用 $x^{(0)}$ 代替 $\frac{dx^{(1)}}{dt}$, 用 $z^{(1)}$ 代替 $x^{(1)}$,

其中

$$z^{(1)}(k) = 0.5 \times x^{(1)}(k) + 0.5 \times x^{(1)}(k-1), \quad k=2, 3, \dots, n \tag{5}$$

式(3)可表示为

$$x^{(0)}(k) + az^{(1)}(k) = b \tag{6}$$

式(6)称为 GM(1, 1)模型的基本形式. 其中, $-a$ 称为发展系数, b 称为灰色作用量. 式(6)的解为

$$\hat{x}^{(1)}(k+1) = \left(x^{(0)}(1) - \frac{b}{a}\right)e^{-ak} + \frac{b}{a}, \quad k=1, 2, \dots, n \tag{7}$$

其中参数 a, b 可用最小二乘法使曲线和累加序列点达到最佳拟合来确定. 令 $\hat{a} = [a \quad b]^T$, 且

$$Y_N = \begin{bmatrix} x^{(0)}(2) \\ x^{(0)}(3) \\ \vdots \\ x^{(0)}(n) \end{bmatrix}, \quad B = \begin{bmatrix} -z^{(1)}(2) & 1 \\ -z^{(1)}(3) & 1 \\ \vdots & \vdots \\ -z^{(1)}(n) & 1 \end{bmatrix},$$

式(6)的参数满足 $\hat{a} = [a \quad b]^T = (B^T B)^{-1} B^T Y_N$, 可得到 a, b 的解.

第 3 步. 预测. 把 a, b 的值代入式(7), 并令 $k=N$ 得

$$\hat{x}^{(1)}(N+1) = \left(x^{(0)}(1) - \frac{b}{a}\right)e^{-aN} + \frac{b}{a} \tag{8}$$

得到累加序列的第 $N+1$ 个点的值, 再累减还原得原始序列第 $N+1$ 个点的预测值:

$$\hat{x}^{(0)}(N+1) = \hat{x}^{(1)}(N+1) - \hat{x}^{(1)}(N) \tag{9}$$

3.2 Verhulst 模型的预测过程

Verhulst 模型是一个变量二阶微分方程, 用来模拟饱和型的数列, 特点是序列单调变化, 但变化率逐渐减小趋近于 0. 其方程为

$$X^{(0)} + aZ^{(1)} = b(Z^{(1)})^2 \tag{10}$$

其中, $Z^{(1)}$ 的定义如式(5). 预测过程与 GM(1, 1)模型类似, 不同的是 GM(1, 1)模型对累加数列 $X^{(1)}$ 进行模拟, Verhulst 直接模拟原始序列, 即把原始序列作为 $X^{(1)}$ ^[22]. 式(10)的解为

$$\hat{x}^{(1)}(k+1) = \frac{ax^{(1)}(0)}{bx^{(1)}(0) + (a - bx^{(1)}(0))e^{ak}}, \quad k=1, 2, \dots, n \tag{11}$$

其中, 参数 a, b 的值可通过使方程与建模序列间满足最小二乘法来得到, 即 $\hat{a} = [a \quad b]^T = (B^T B)^{-1} B^T Y_N$, 其中

$$Y_N = \begin{bmatrix} x^{(0)}(2) \\ x^{(0)}(3) \\ \vdots \\ x^{(0)}(n) \end{bmatrix}, \quad B = \begin{bmatrix} -z^{(1)}(2) & (z^{(1)}(2))^2 \\ -z^{(1)}(3) & (z^{(1)}(3))^2 \\ \vdots & \vdots \\ -z^{(1)}(n) & (z^{(1)}(n))^2 \end{bmatrix},$$

把 a 、 b 的值代入式 (11), 可计算出原始序列的预测值.

3.3 基于灰色模型的软件阶段成本预测

灰色模型的优势是可以少量数据建模, 通常数据越多建模效果越好, 不过, 这需要序列整体比较光滑, 变化不能太剧烈, 否则过多的数据反而不能及时反映数据的变化. 对软件阶段成本序列来说, 数据有时变化比较剧烈, 有时变化比较缓慢, 总体来说变化比较频繁. 针对这种情况, 我们不使用灰色模型通常的作法, 尽可能多地使用现有数据构造预测模型, 而是只选择最近 3 个阶段的数据建模, 达到反映数据变化趋势的目的, 以便获得较高的预测精度. GM(1,1)模型是一种准指数模型, 当建模序列变化比较剧烈时, 其预测误差常常大到无法接受的程度, 因此适合预测变化缓慢的序列; Verhulst 模型用于模拟饱和型的序列, 即使建模数列波动较大, 预测误差也不会失去控制, 反而对变化过于缓慢的序列会出现模型参数无效的问题. 序列变化的剧烈程度可

以用变化率来表示. 根据软件阶段成本序列和两种灰色模型的特点, 可以考虑在预测时先确定建模子序列的变化率, 对变化率大的序列用 Verhulst 模型来建模, 变化率小的用 GM(1,1)模型建模, 通过两种模型的优势互补获得更高的预测精度. 由此我们提出一种利用成本序列及其变化率动态构造灰色模型进行软件阶段成本预测的方法, 称之为 GMCP (Grey Model for software Cost Prediction)方法.

GMCP 方法的基本过程为

- 1. 从软件历史项目成本数据中学习得到用于判定序列变化大小的变化率阈值;
- 2. 根据待预测成本序列变化率选择模型进行预测;
- 3. 通过对历史项目进行预测, 总结出方法系统偏差大小, 对预测值进行误差补偿, 优化预测过程.

对每一次具体的预测过程, 先要进行数据预处理, 例如进行光滑性检查, 必要时进行数据变换, 然后再预测. 在真实数据集上进行预测的过程如图 1 所示.

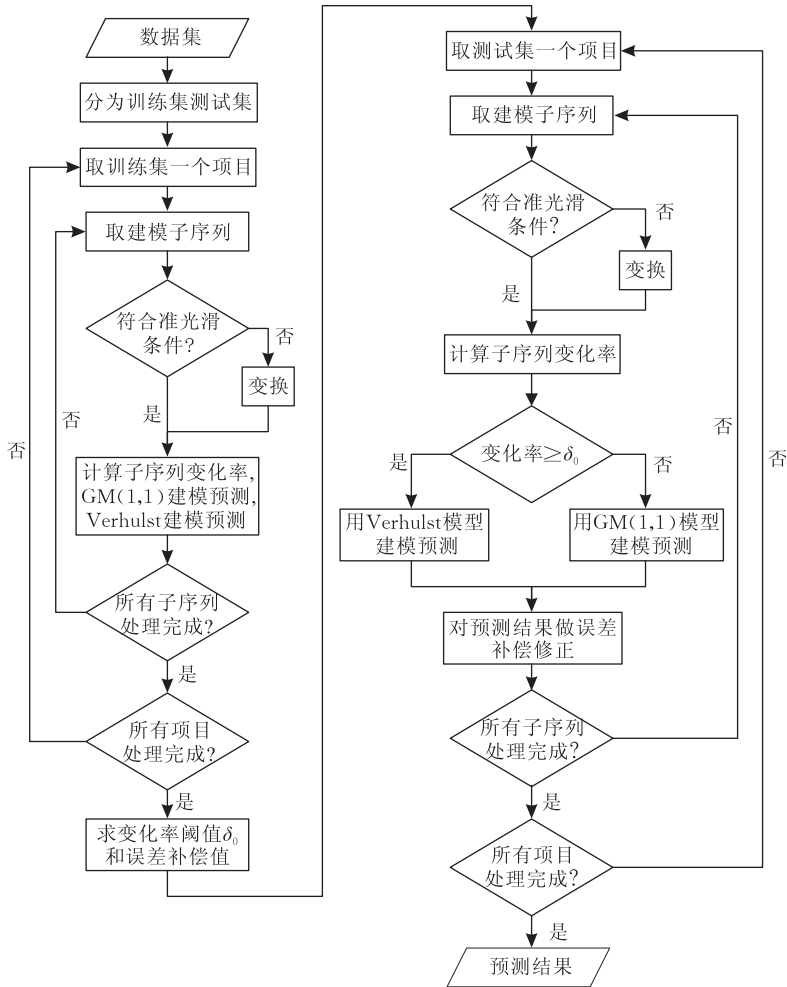


图 1 GMCP 方法的实现过程

下面分别介绍数据预处理、求序列变化率阈值和误差补偿值、预测并误差补偿的实现过程。

(1)数据预处理. 数据预处理的目的是减少原始序列的离散性, 增加光滑性, 使序列适合建立灰色模型. 如果用离散的原始序列直接建模, 一般不容易拟合而造成预测误差很大. 通过检验原始序列的光滑性, 确定序列是否适合建模. 假设 $X^{(0)} = (x^{(0)}(1), x^{(0)}(2), \dots, x^{(0)}(n))$ 是原始序列, 序列光滑性的定义如下^[20,22]:

$$\sigma(k) = \frac{x^{(0)}(k)}{x^{(0)}(k-1)}, \quad k=2, \dots, n \tag{12}$$

当 $\sigma(k) \in [e^{-\frac{2}{n+1}}, e^{\frac{2}{n+1}}]$ 时, 称序列 $X^{(0)}$ 是准光滑的, 可以用来建立灰色模型.

如果序列不符合准光滑条件, 则需要做变换处理. 这种处理不仅要使原始序列变得光滑, 而且应保留序列变化的趋势信息, 以便提取序列变化率, 作为判定序列变化快慢的依据. 确定趋势的常见方法是 n 阶的加权移动平均 (weighted moving average of order n)^[33]. 设 $X = (x_1, x_2, x_3, \dots)$ 为给定数列, 可按如下的算术平均式形式, 计算 k 阶的加权移动平均值:

$$\left\{ \begin{aligned} &\frac{\omega_1 x_1 + \omega_2 x_2 + \dots + \omega_k x_k}{\omega_1 + \omega_2 + \dots + \omega_k}, \\ &\frac{\omega_2 x_2 + \omega_3 x_3 + \dots + \omega_{k+1} x_{k+1}}{\omega_2 + \omega_3 + \dots + \omega_{k+1}}, \\ &\frac{\omega_3 x_3 + \omega_4 x_4 + \dots + \omega_{k+2} x_{k+2}}{\omega_3 + \omega_4 + \dots + \omega_{k+2}}, \\ &\dots \end{aligned} \right. \tag{13}$$

加权移动平均可以降低序列的变化, 减少不希望出现的波动, 但移动平均会丢失数列中的头尾数据. 软件阶段成本数据通常很少, 如果去掉头尾数据, 数据更少, 可能无法建模. 因此, 对式(13)的算法进行改进, 增加头尾数据. 例如, 3 阶加权移动平均, 增加头尾数据后的序列为

$$\frac{2x_1 + x_2}{3}, \dots, \frac{x_{k-1} + x_k + x_{k+1}}{3}, \dots, \frac{x_{n-1} + 2x_n}{3} \tag{14}$$

尾部数据最靠近被预测点, 加重权值可以突出其重要性, 同时使变换后的序列仍比较光滑.

对于数列变化率, 可以用灰色系统理论中的级比偏差来表示^[34]. 设 $X = (x(1), x(2), \dots, x(n))$ 为数列, 其级比偏差值可用下式表示:

$$\delta = \left| \frac{x(k+1) - x(k)}{x(k)} \right| = \left| \frac{\Delta x(k)}{x(k)} \right|, \quad k=2, 3, \dots, n \tag{15}$$

式中 δ 反映了数列相邻两元素相对变化的大小, 数据增加或减少得越多, δ 的值越大, 数据变化很小时, δ 趋于零. 对上述定义作改造, 可得表示成本子序列变化的级比偏差值:

$$\delta = \left| \frac{x(m+k-1) - x(m)}{x(m)} \right| \tag{16}$$

其中, k 代表序列元素的个数, $x(m+k-1)$ 是序列最后一个元素, $x(m)$ 是第一个元素. 本文中称式(16)中的 δ 为序列的变化率, δ 越大, 表示序列变化越快, 反之越慢.

(2)求变化率阈值和误差补偿值. 基本思想是, 在所有建模序列的变化率集合中存在一个变化率阈值 δ_0 , 用它作为判断序列变化快慢的标准进而选择预测模型得到的总平均预测误差最小; 所有预测偏差的均值反映了数据集总体的预测偏差, 可作为误差补偿值来修正未来预测的值. 通常在训练集上求 δ_0 和误差补偿值. 图 2 给出了实现过程的伪代码.

```
1. m=建模子序列个数;
2. for 每个项目 prj∈训练集 {
3.   for 项目每个建模子序列 SubX(0)∈prj {
4.     if SubX(0) 不满足光滑性条件 then
5.       SubX(0) = 经过光滑变换的 SubX(0);
6.       //求序列的变化率及两种模型预测结果
7.       δ(i) = SubX(0) 的变化率;
8.       MREGM(i) = SubX(0) 用 GM(1,1) 模型预测的误差;
9.       BiasGM(i) = SubX(0) 用 GM(1,1) 模型预测的偏差;
10.      MREVER(i) = SubX(0) 用 Verhulst 模型预测的误差;
11.      BiasVER(i) = SubX(0) 用 Verhulst 模型预测的偏差;
12.    }
13.  }
14. M = (δ, MREGM, BiasGM, MREVER, BiasVER);
    //预测结果合并
15. Sort(M by δ); //把矩阵 M 按 δ 从小到大排序
16. for 每个 δi in M
17.   MMRE(δi) = Mean(Sum(MREGM(1:i)) + Sum(MREVER(i+1:N)));
18. δ0 = 使 MMRE 最小的 δi //得到变化率阈值
19. I = δi 在 M 中的行号
20. Bias0 = Mean(BiasGM(1:I) + BiasVER(I+1:N))
    //误差补偿值
```

图 2 求成本序列变化率阈值 δ_0 和误差补偿值过程

首先指定建模子序列个数 m , 即用项目多少个阶段的数据建立预测模型 (第 1 行), 从数据集中取出一个项目的阶段成本序列 (第 2 行), 从中取出 m 个元素构成建模子序列 (每取一次, 起点后移 1 个阶段) (第 3 行), 检查序列是否光滑, 如不光滑则进行变换 (第 4、5 行), 变换的方法如式(13); 然后分别计算序列的变化率 δ 、用 GM(1,1) 模型和 Verhulst 模型建模预测的相对预测误差及偏差 (第 7~11 行); 把所有序列的预测结果按 δ 从小到大排序 (第 15 行), 取出一个 δ 作为阈值, 用它作为选择模型的标

准,计算数据集所有子序列的平均预测误差(第 16、17 行)(其中 $MRE_{GM}(1:i)$ 表示返回第 1 个到第 i 个元素);找出使平均预测误差 $MMRE$ (详见第 4.2 节)最小的 δ ,该 δ 即变化率阈值 δ_0 (第 18 行);从预测结果中取出该 δ_0 对应的各预测值的偏差,取这些偏差均值作为误差补偿值(第 19、20 行).

(3) 预测并进行误差补偿. 把建模子序列的变化率 δ 和变化率阈值 δ_0 比较,如果 $\delta < \delta_0$,用 $GM(1,1)$ 模型预测;如果 $\delta \geq \delta_0$,用 Verhulst 模型预测,用误差补偿值修正得到的预测值. 实现过程的伪代码如下所示. 首先,从数据集中取出一个项目工作量序列(第 2 行);再取出其中 m 个阶段构成建模子序列(第 3 行);判断序列是否光滑并处理(第 4、5 行);求序列的变化率 δ (第 6 行),根据和 δ_0 比较的结果选择模型预测,用训练时得到的误差补偿值 $Bias_0$ 修正预测值(第 8、10 行),计算预测误差(第 11 行);各子序列预测误差的平均值作为该项目的预测误差(第 13 行);所有项目预测误差的均值作为数据集的预测误差值(第 15 行).

```
1.   $m$ =建模子序列个数
2.  for 每个项目  $prj \in$  测试集 {
3.      for 项目每个建模子序列  $SubX^{(0)} \in prj$  {
4.          if  $SubX^{(0)}$  不满足光滑性条件 then
5.               $SubX^{(0)}$  = 经过光滑变换的  $SubX^{(0)}$ ;
//对序列作光滑变换
6.           $\delta$ = $SubX^{(0)}$  的变化率;
7.          if  $\delta \geq \delta_0$  then
8.              预测值=Verhulst 模型预测值/ $(1-Bias_0)$ ;
9.          else
10.             预测值= $GM(1,1)$ 模型预测值/ $(1-Bias_0)$ ;
11.              $MRE(j)=SubX^{(0)}$  的相对预测误差;
12.         }
13.      $MRE_{prj}(i)=Mean(MRE)$  //项目预测误差
14. }
15.  $MMREPRJ=Mean(MRE_{prj})$  //数据集相对误差平均值
```

图 3 预测及误差补偿过程

4 实验与结果

4.1 数据集

本实验所用的数据集来自 EDS 公司维护的一个软件项目数据仓库,构建这个数据仓库的主要目的是为软件企业和研究机构提供软件评估、度量和预测方面的支持,这个数据仓库在一定程度上解决了软件工程研究领域数据集较少的问题. 实验数据集共包括 4390 个软件项目,每个项目包含了丰富的属性信息,如行业、项目类型、开发组、开发平台、开发语言等. 与成本估计研究中常用的数据集相比,这

个数据仓库包含的项目数量较大,内容更加丰富. 项目实施的地点包括美国(56.7%)、澳大利亚(7%)、瑞士(5%)、加拿大(4.6%)、新西兰(4.2%)、英国(3.9%)等 30 多个国家和地区;应用的行业有航天、金融、制造、医疗、零售、交通等;使用的计算机语言有 COBOL(10.9%)、JCL(9.4%)、SQL(5.2%)、PL/1(5%)、Visual Basic(4.5%)、Assembler(4.3%)、C(3.5%)、Java(2.9%)、C++(2.2%)等 130 多种. 通过复杂的数据结构和严格完备的度量方法,项目信息被完整的记录下来,其中成本记录尤其详尽. 开发过程不是按生命周期阶段简单分类,而是根据项目类型、开发方法定义了一套活动(activity)分级目录,包含了软件项目中出现的约一千多种类型的活动,实际项目包含的活动都是目录中活动的子集,各项活动的成本逐月记录. 由于不同项目所包含活动的类型和数量各不相同,为了便于分析阶段成本的演化,把同一个项目同一个月的各项活动成本汇总起来,生成以月为间隔,以项目持续月数为长度的成本数据序列,以此研究软件阶段成本的演化规律和预测问题. 研究发现,不同行业的项目存在较大差异,为了使本文所提方法对行业有更好的适用性,我们把所有项目根据行业分成 10 个数据集,分别进行实验. 表 1 和表 2 描述了数据集的基本情况.

表 1 项目所在的行业分布

ID	项目数	行业
1	100	Aerospace
2	42	Distribution
3	967	Financial
4	2436	Manufacturing
5	74	Health
6	80	Retail
7	58	Service
8	438	Transportation
9	113	Systems Integration
10	82	Other

表 2 项目工作量(人小时)统计描述

ID	Mean	Median	Min	Max	Std. Dev
1	5207.35	1688.88	3.00	86184.00	11018.02
2	15048.85	10345.13	64.00	66814.25	14798.67
3	9283.49	2248.75	0.00	263848.25	20349.78
4	7215.58	2469.38	0.00	212314.50	14707.49
5	31780.58	9409.50	28.50	420408.00	64404.46
6	18027.77	8129.88	215.75	167569.75	28544.98
7	17407.13	3295.25	18.50	119690.50	31095.64
8	8253.70	3073.50	0.00	132273.50	14770.07
9	16639.02	9579.25	77.25	116681.75	19881.69
10	11644.06	5647.50	0.00	74355.00	15704.62

如表 1 所示,10 个数据集分属 10 个不同的行业,其中项目数最少的是 42 个,最多的是 2436 个,平均

439 个。从表 2 可以看出,同一数据集中项目的工作量差别很大,标准差介于 11018.02~64404.46 之间,最大值达到 420408 人小时,平均值介于 5205~31780 人小时之间,中位数介于 1688~10345 人小时之间,其中最小值为 0 的属无效记录,将在实验时删除。

项目持续时间情况如图 4 所示。在 4390 个项目中,有 421 个项目的持续时间小于或等于 3 个月,占全部记录数的 9.6%,其它项目绝大多数在 6~27 个月之间,所有项目平均工期约 15 个月。本实验需要用连续 3 个月的成本数据建模,预测第 4 个月的成本,因此,小于或等于 3 个月的项目因数据太少,不能建模或无法检验预测精度,将从数据集中删掉。总体来说,绝大多数项目至少可以使用本文方法预测余下一半工期的工作量,项目平均可预测工作量占总工作量的 80%。因此,对大多数项目来说,在中前期就可以预测后续阶段成本,从而有充分的时间对原计划进行改进。另外,符合条件的项目总数接近 4000 个,且具有足够的多样性,使在此基础上得到的预测方法有更好的适用性和普遍性。

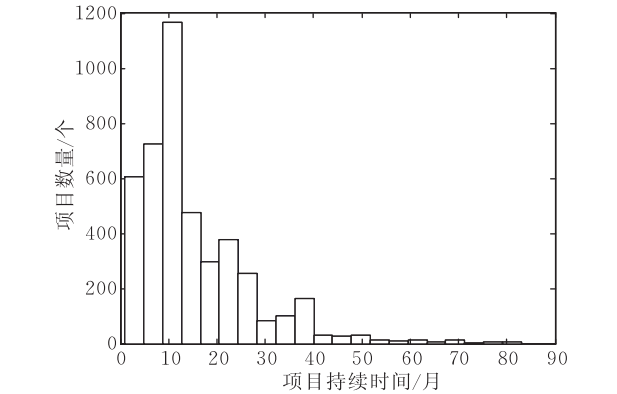


图 4 项目阶段数分布图

在实验中使用保持法 (holdout) 来设计实验数据集,保持法在大的数据集上可以节省计算时间。软件阶段成本数据趋势变化较快,为了反映变化信息,使用连续 3 个月成本数据建模,预测后续一个月的成本。对不满足式 (12) 光滑性条件的建模序列,用式 (14) 中的方法进行变换。为了比较 GMCP 方法的性能,用线性回归方法 (LinR) 作为基准方法在所有测试集上进行对比实验,为保证公平,两种方法所用的数据完全相同。

4.2 评价标准

在软件成本估计领域有 4 种常用的评价指标,分别是 *Bias*、*MMRE*、*MdMRE* 和 *PRED*,通常在研究中使用其中一种或几种。本研究为了充分检验所提方法的性能,使用全部 4 种指标来评估预测的结果。

Bias 表示预测值与实际值的偏离程度,对被预测值 *i*,对应的定义如下:

Bias_i = (x_i - x_i_hat) / x_i * 100% (17)

其中, *x_i* 为实际值, *x_i*[^] 为预测值,该值越接近 0 表示预测精度越高。

相对误差 (Magnitude of Relative Error, *MRE*) 通常用来评价预测方法的精度,定义如下:

MRE_i = |x_i - x_i_hat| / x_i * 100% (18)

其中 *x_i*[^] 是实际值 *x_i* 的预测值。

对多次预测的平均相对误差 (Mean Magnitude of Relative Error, *MMRE*) 有如下定义:

MMRE = 1/n * sum(MRE_i) (19)

MMRE 是最常用的预测误差的评价指标,但它对 *MRE* 中的极值敏感,通常取 *MRE* 的中位数 *MdMRE* 作为另一个评价指标,以消除极值的影响。对 *MMRE* 和 *MdMRE* 来说,值越大代表预测精度越低。

另一个常用的衡量预测质量的指标是 *PRED* (*m*), 它表示 *MRE* 小于 *m*% 的预测个数占总预测个数的百分比,定义为

PRED(m) = k/n * 100% (20)

其中 *k* 表示 *MRE* ≤ *m*% 的预测个数, *n* 表示全部预测个数,通常 *m* 取 25。 *PRED* 越大表示精度高的预测所占比例越大,但它对 *m*% 以外的预测不敏感,例如,一个项目的 *MRE* 是 26% 或 260% 对 *PRED* (25) 取值没有影响。

4.3 实验结果与分析

用 GMCP 依照本文 3.3 节的步骤在 10 个数据集上进行实验,同时用 LinR 进行相应的实验。两种方法所得到的 *MMRE*、*MdMRE*、*PRED* (25) 和 *Bias* 4 项指标的值,分别列于表 3 和表 4 中。

表 3 GMCP 和 LinR 在不同数据集上的 MMRE 和 MdMRE

ID	MMRE/%		MdMRE/%	
	GMCP	LinR	GMCP	LinR
1	47.20	72.30	25.48	35.06
2	21.54	29.79	17.70	29.04
3	55.71	78.25	23.92	29.15
4	101.27	162.68	28.55	38.11
5	119.43	217.05	17.77	21.69
6	23.68	30.99	17.85	23.44
7	21.50	28.28	22.58	28.53
8	30.19	36.39	20.28	26.30
9	17.56	22.63	13.22	17.00
10	21.73	30.97	17.70	22.05

表 4 GMCP 和 LinR 在不同数据集上的 $PRED(25)$ 和 $Bias$

ID	$PRED(25)/\%$		$Bias/\%$	
	GMCP	LinR	GMCP	LinR
1	48.00	24.00	-4.46	-34.32
2	64.29	42.86	7.72	-6.25
3	51.66	43.13	-12.87	-13.64
4	42.33	29.07	-60.57	-76.49
5	63.64	54.55	-94.27	-131.00
6	75.00	54.17	-1.53	-5.51
7	64.29	28.57	12.19	-4.41
8	58.27	45.67	-1.21	-8.56
9	87.50	78.13	1.17	-5.00
10	82.61	56.52	7.31	-6.16

从表 3 可以看出,在 10 个数据集上,GMCP 在 $MMRE$ 和 $MdMRE$ 两个指标上的表现全部超过 LinR. 相比 LinR,GMCP 在 $MMRE$ 上最少降低了 20.54%,在 $MdMRE$ 上最少降低了 21.86%. 在数据集 4、5 上两种模型所得的 $MdMRE$ 都较小,而 $MMRE$ 都较大,说明数据集中存在极值点. 同时,在这两个数据集上,GMCP 的 $MMRE$ 相比 LinR 减小的也最多,分别为 60.64%和 81.74%,表明 GMCP 比 LinR 有更强的处理极值点的能力.

由表 4 可以看出,GMCP 在所有 10 个数据集上的 $PRED(25)$ 指标全部优于 LinR,最少提高了 10.71%,说明 GMCP 预测精度高的比例要大. 对于 $Bias$ 指标,GMCP 在 7 个数据集上占优,在两个数据集上取得与 LinR 相近的值,一个数据集上的取值比 LinR 差. GMCP 的预测结果按照 3.3 节的方法进行过误差补偿,从数值上看,GMCP 比 LinR 的 $Bias$ 更接近 0,说明误差补偿产生了效果. 同时注意到两个模型的 $Bias$ 绝大多数是负值,说明结果偏向过估计,其中 3 个数据集上的 $Bias$ 值较大,说明预测偏差在数据集间的分布不均匀. 因此,还可以进一步改进预测及误差补偿方法.

表 3 数据集 4、5 上的 $MMRE$ 值均较大,检查发现原始数据序列存在异常波动,使预测误差显著增大,造成整个数据集预测精度大幅下降. 因此,提高原始数据的质量是软件阶段成本预测工作的一个重要前提.

总之,在本文使用的 10 个数据集上,GMCP 在 $MMRE$ 、 $MdMRE$ 、 $Pred(25)$ 和 $Bias$ 4 项指标上,取得了较好的预测结果,均超过 LinR.

5 结 论

软件阶段成本预测可以帮助管理人员预见项目潜在的问题,对保证项目在进度和预算内完成有重

要意义. 本文在对灰色模型、软件阶段成本数据等深入分析的基础上,提出一种基于灰色模型的软件项目阶段成本预测方法,并在 10 个真实世界软件工程数据集上进行了验证. 结果表明,该方法明显优于线性回归方法,有较高的预测精度,显示出较大的潜力.

致 谢 在本文工作的研究过程中,作者曾多次和英国 Brunel 大学的 Shepperd 教授讨论,Shepperd 教授提出了很多有益的建议,师兄翁小清博士对实验结果的统计分析提供了大量指导,在此向他们表示衷心的感谢! 作者也感谢为本文研究提供数据的 EDS 公司!

参 考 文 献

[1] Jones T C. Estimating Software Costs. New York: McGraw Hill, 1998

[2] Moløken K, Jørgensen M. A review of surveys on software effort estimation//Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE 2003). Rome, Italy, 2003: 223-230

[3] Wang Y, Song Q, Shen J. Grey learning based software stage-effort estimation//Proceedings of the 6th International Conference on Machine Learning and Cybernetics. Hong Kong, China, 2007: 1470-1475

[4] Wang Y, Song Q, Shen J. Grey prediction based software stage-effort estimation. Wuhan University Journal of Natural Sciences, 2007, 12(5): 927-931

[5] Glass R L. Facts and Fallacies of Software Engineering. Boston: Addison-Wesley, 2003

[6] Briand L C, Langley T, Wiecezorek I. A replicated assessment and comparison of common software cost modeling techniques//Proceedings of the 22th International Conference on Software Engineering (ICSE 2000). Limerick, Ireland, 2000: 377-386

[7] Hughes R T. Expert judgment as an estimating method. Information & Software Technology, 1996, 38(2): 67-75

[8] Shepperd M, Schofield C. Estimating software project effort using analogies. IEEE Transactions on Software Engineering, 1997, 23(11): 736-743

[9] Putnam L H. Example of an early sizing, cost and schedule estimate for an application software system//Proceedings of the 2nd International Computer Software and Applications Conference (COMPSAC 1978). Chicago, USA, 1978: 827-832

[10] Srinivasan K, Fisher D. Machine learning approaches to estimating software development effort. IEEE Transactions on Software Engineering, 1995, 21(2): 126-137

- [11] Briand L C, Emam K E, Surmann D, Wiczorek I, Maxwell K D. An assessment and comparison of common software cost estimation modeling techniques//Proceedings of the 1999 International Conference on Software Engineering (ICSE 1999). Los Angeles, USA, 1999: 313-322
- [12] He Xiao-Yang, Wang Ya-Sha. Model-based methods for software cost estimation. Journal of Computer Research and Development, 2006, 43(5): 777-783(in Chinese)
(何晓阳, 王亚沙. 基于模型的软件成本估计方法. 计算机研究与发展, 2006, 43(5): 777-783)
- [13] Li Ming-Shu, He Mei, Yang Da, Shu Feng-Di, Wang Qing. Software cost estimation method and application. Journal of Software, 2007, 18(4): 775-795(in Chinese)
(李明树, 何梅, 杨达, 舒凤笛, 王青. 软件成本估算方法及应用. 软件学报, 2007, 18(4): 775-795)
- [14] Jørgensen M, Shepperd M. A systematic review of software development cost estimation studies. IEEE Transactions on Software Engineering, 2007, 33(1): 33-53
- [15] Kemerer C F. An empirical validation of software cost estimation models. Communications of the ACM, 1987, 30(5): 416-429
- [16] Boehm B W, Fairley R E. Software estimation perspectives. IEEE Software, 2000, 17(6): 22-26
- [17] Boehm B W, Clark, Horowitz, Brown, Reifer, Chulani, Madachy R, Steece B. Software Cost Estimation with CO-COMO II. New York: Prentice Hall, 2000
- [18] Srinivasan K, Fisher D. Machine learning approaches to estimating software development effort. IEEE Transactions on Software Engineering, 1995, 21(2): 126-137
- [19] Idri A, Khoshgoftaar T M, Abran A. Can neural networks be easily interpreted in software cost estimation?//Proceedings of the 2002 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'02). Hawaii, USA, 2002: 1162-1167
- [20] Deng Ju-Long. The Course of Grey System Theory. Wuhan: Huazhong University of Science and Technology Press, 1990 (in Chinese)
(邓聚龙. 灰色系统理论教程. 武汉: 华中科技大学出版社, 1990)
- [21] Deng J. Control problems of grey systems. Systems & Control Letters, 1982, 1(5): 288-294
- [22] Liu Si-Feng, Dang Yao-Guo, Fang Zhi-Geng. The Grey System Theory and Application. 3rd Edition. Beijing: Science Press, 2004(in Chinese)
(刘思峰, 党耀国, 方志耕. 灰色系统理论及应用. 第3版. 北京: 科学出版社, 2004)
- [23] Deng J. Introduction to grey system theory. The Journal of Grey System, 1989, 1(1): 1-24
- [24] El-Fouly T H M, El-Saadany E F, Salama M M A. Grey predictor for wind energy conversion systems output power prediction. IEEE Transactions on Power Systems, 2006, 21(3): 1450-1452
- [25] Su S L, Su Y C, Huang J F. Grey-based power control for DS-CDMA cellular mobile systems. IEEE Transactions on Vehicular Technology, 2000, 49(6): 2081-2088
- [26] Lin K, Liu B. A gray system modeling approach to the prediction of calibration intervals. IEEE Transactions on Instrumentation and Measurement, 2005, 54(1): 297-304
- [27] Jou J M, Chen P Y, Sun J M. The gray prediction search algorithm for block motion estimation. IEEE Transactions on Circuits and Systems for Video Technology, 1999, 9(6): 843-848
- [28] Song Q, Shepperd M, Mair C. Using grey relational analysis to predict software effort with small data sets//Proceedings of the 11th International Software Metrics Symposium (METRICS 2005). Rome, Italy, 2005: 35
- [29] MacDonell S, Shepperd M. Using prior-phase effort records for re-estimation during software projects//Proceedings of the 9th International Software Metrics Symposium (METRICS 2003). Sydney, Australia, 2003: 73-86
- [30] Kulkarni A, Greenspan J B, Kriegman D A, Logan J J, Roth T D. A generic technique for developing a software sizing and effort estimation model//Proceedings of the 12th International Computer Software and Applications Conference (COMPSAC 1988). Chicago, USA, 1988: 151-161
- [31] Ohlsson M, Wohlin C. An empirical study of effort estimation during project execution//Proceedings of the 6th International Software Metrics Symposium (METRICS 1999). Boca Raton, USA, 1999: 91-98
- [32] Boehm B W. Software Engineering Economics. Englewood Cliffs: Prentice-Hall, 1981
- [33] Han J W, Kamber M. Data Mining: Concepts and Techniques. San Francisco: Morgan Kaufmann, 2001
- [34] Deng Ju-Long. Grey Prediction and Grey Decision-Making. Wuhan: Huazhong University of Science and Technology Press, 2002(in Chinese)
(邓聚龙. 灰预测与灰决策. 武汉: 华中科技大学出版社, 2002)



WANG Yong, born in 1971, Ph.D.. His research interests include software engineering economics, data mining and software engineering.

SONG Qin-Bao, born in 1966, professor, Ph.D. supervisor. His research interests include data mining and software engineering, trustworthy software.

SHEN Jun-Yi, born in 1939, professor, Ph.D. supervisor. His research interests include data mining, database system theory.

Background

The software stage effort estimation plays an important role in the area of the software project management. There are about 75% of the projects all over the world overrun the schedules because of the inaccurate software cost prediction. In practice the software projects are often out of control, can't be finished on budget and with poor quality for the absence of effective project cost prediction, and it is difficult to enter the international software market.

Although there are several software cost prediction methods, however, no one method is consistently effective. Without considered the special characters of software development process, these methods can not deal with the uncertain data, and they require a large data samples with certain distribution. The Grey System Theory (GST) based on the uncertainty of small samples, doesn't require a typical distri-

bution, and has been successfully and widely applied in many areas. Prof. Song, one of the authors of this paper, and Prof. Shepperd et al. have used this theory successfully addressed the whole project cost estimation problem. Encouraged by their job, in this research the authors focus on the software stage effort evolution and estimation questions during the development process by the GST. The experimental results show that, compared with the current method, the prediction accuracy of the proposed method has been improved by 20%~80%.

The research is supported by the National Natural Science Foundation of China (60673124, 61673087, 90718024) and the National High Technology Research and Development Program (863 Program) of China (2006AA01Z183).