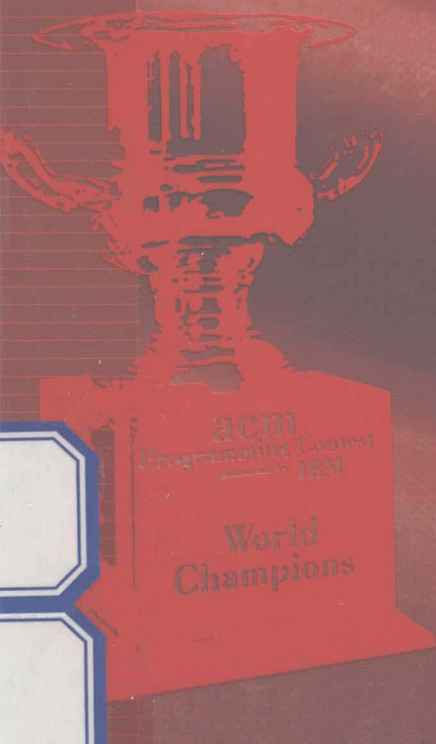


谨以此书献给上海交通大学获得ACM-ICPC世界冠军十周年

ACM国际大学生程序设计竞赛 (ACM-ICPC) 系列丛书

ACM国际大学生 程序设计竞赛 算法与实现

俞 勇 主编



清华大学出版社



ACM国际大学生程序设计竞赛 (ACM-ICPC) 系列丛书

ACM国际大学生 程序设计竞赛 算法与实现

俞 勇 主编



清华大学出版社
北 京

内 容 简 介

ACM 国际大学生程序设计竞赛 (ACM-ICPC) 是国际上公认的水平最高、规模最大、影响最深的计算机专业竞赛, 目前全球参与人数达 20 多万。本书作者将 16 年的教练经验与积累撰写成本系列丛书, 全面、深入而系统地将 ACM-ICPC 展现给读者。本系列丛书包括《ACM 国际大学生程序设计竞赛: 知识与入门》、《ACM 国际大学生程序设计竞赛: 算法与实现》、《ACM 国际大学生程序设计竞赛: 题目与解读》、《ACM 国际大学生程序设计竞赛: 比赛与思考》等 4 册, 其中《ACM 国际大学生程序设计竞赛: 知识与入门》介绍了 ACM-ICPC 的知识及其分类、进阶与角色、在线评测系统; 《ACM 国际大学生程序设计竞赛: 算法与实现》介绍了 ACM-ICPC 算法分类、实现及索引; 《ACM 国际大学生程序设计竞赛: 题目与解读》为各类算法配备经典例题及题库, 并提供解题思路; 《ACM 国际大学生程序设计竞赛: 比赛与思考》介绍了上海交通大学 ACM-ICPC 的训练及比赛, 包括训练札记、赛场风云、赛季纵横、冠军之路、峥嵘岁月。

本丛书适用于参加 ACM 国际大学生程序设计竞赛的本科生和研究生, 对参加青少年信息学奥林匹克竞赛的中学生也很有指导价值。同时, 作为程序设计、数据结构、算法等相关课程的拓展与提升, 本丛书也是难得的教学辅助读物。

本书封面贴有清华大学出版社防伪标签, 无标签者不得销售。

版权所有, 侵权必究。侵权举报电话: 010-62782989 13701121933

图书在版编目 (CIP) 数据

ACM 国际大学生程序设计竞赛: 算法与实现/俞勇主编. —北京: 清华大学出版社, 2013.1

ACM 国际大学生程序设计竞赛 (ACM-ICPC) 系列丛书

ISBN 978-7-302-29413-9

I. ①A… II. ①俞… III. ①程序设计—竞赛—高等学校—教学参考资料 IV. ①TP311.1

中国版本图书馆 CIP 数据核字 (2012) 第 161172 号

责任编辑: 龙启铭

封面设计: 傅瑞学

责任校对: 李建庄

责任印制: 何 芊

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮编: 100084

社 总 机: 010-62770175 邮购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者: 北京国马印刷厂

经 销: 全国新华书店

开 本: 185mm×230mm

印 张: 17.75

字 数: 445 千字

版 次: 2013 年 1 月第 1 版

印 次: 2013 年 1 月第 1 次印刷

印 数: 1~3000

定 价: 36.00 元

产品编号: 047607-01

自从上海交通大学 2002 年第一次、2005 年第二次获得 ACM 国际大学生程序设计竞赛 (ACM International Collegiate Programming Contest, 简称 ACM-ICPC 或 ICPC) 世界冠军以来, 总有记者邀请编者撰写冠军之路类的文章, 也总有出版社希望编者出版 ACM-ICPC 竞赛类的书籍, 因为没有想清楚怎么写, 所以一直没动笔。直到 2010 年上海交通大学第三次获得 ACM-ICPC 世界冠军后, 编者决定出版一套系列丛书, 包括《ACM 国际大学生程序设计竞赛: 知识与入门》、《ACM 国际大学生程序设计竞赛: 算法与实现》、《ACM 国际大学生程序设计竞赛: 题目与解读》及《ACM 国际大学生程序设计竞赛: 比赛与思考》4 册书籍, 全面、深入而系统地将 ACM-ICPC 展现给读者, 把上海交通大学十多年来对 ACM-ICPC 竞赛的感悟分享给读者。

编写此系列丛书的另一个重要原因是 ACM-ICPC 竞赛在中国大陆的迅猛发展。自从 1996 年 ACM-ICPC 引入中国大陆, 前六届仅设立 1 个赛区, 目前每年一般设立 5 个赛区, 并已有 30 所高校承办过亚洲区预赛; 参赛学校从不满 20 所, 到如今已达 200 多所; 参赛人数从不到 100 人, 到如今超过 12 万人次; 总决赛名额从起初的 3 个, 到如今已超过 15 个。同时, 中国大陆在 ACM-ICPC 竞赛上所取得的成绩也举世瞩目。清华大学 9 次获得总决赛奖牌 (3 金 5 银 1 铜), 位居奖牌榜之首, 是实力最强、表现最稳定的高校; 上海交通大学 8 次获得总决赛奖牌 (4 金 3 银 1 铜), 3 次夺得世界冠军, 算是目前国内成绩最好的高校; 中山大学 4 次获得总决赛奖牌 (2 银 2 铜), 在生源不占优势的情况下, 这一成绩令人敬佩; 复旦大学 3 次获得总决赛奖牌 (1 银 2 铜), 是公认的强校; 浙江大学 2 次获得总决赛奖牌 (1 金 1 银), 1 次夺得世界冠军, 再次让国人欢欣鼓舞; 北京大学 1 次获得总决赛奖牌 (1 铜), 队员的综合实力堪称一流; 最难能可贵的是, 华南理工大学也获得过总决赛的奖牌 (1 铜), 它告诉我们, ACM-ICPC 不仅仅是“强校”之间的“对话”, 只要坚持参与就会斩获成果。另外, 至今已有 37 所大陆高校参加过全球总决赛, 且不论成绩如何, 他们在赛场上的奋斗亦值得称道。

本系列丛书的第一册《ACM 国际大学生程序设计竞赛：知识与入门》分为三个部分。知识点部分基本涵盖了竞赛中所涉及的主要知识点，包括数学基础、数据结构、图论、计算几何、论题选编、求解策略等六大类内容。入门与进阶部分介绍了包括如何快速入门、如何提高自身以及团队水平等，主要根据上海交通大学 ACM-ICPC 队多年参赛经验总结而来。在线资源部分对一些常用的在线评测系统和网上比赛进行了介绍。

本系列丛书的第二册《ACM 国际大学生程序设计竞赛：算法与实现》涵盖了大部分 ACM-ICPC 竞赛常用的经典算法，包括数学、图论、数据结构、计算几何、论题选编五大类，对每个算法的代码实现，都配有接口说明以及简略的算法阐述，并提供算法的完整程序，贴士部分收集了一些实用的知识点及积分表，方便读者查找使用。

本系列丛书的第三册《ACM 国际大学生程序设计竞赛：题目与解读》分为两个部分。例题精讲部分针对第二册《ACM 国际大学生程序设计竞赛：算法与实现》中的算法配备经典例题，并提供细致的解题思路，读者可以通过这一部分学习和掌握算法；海量题库部分按照算法分类罗列出大量习题，并提供相应的题解，读者可以利用这一部分的题目进行训练，更加熟练地运用各类算法。

本系列丛书的第四册《ACM 国际大学生程序设计竞赛：比赛与思考》从 120 多名队员、2400 余篇文档中精心挑选、编纂而成的文集，包括训练札记、赛场风云、赛季纵横、冠军之路、峥嵘岁月，集中展现了上海交通大学 ACM-ICPC 队 16 年的奋斗历程，记载了这些队员为了实现自己的梦想而不懈努力、勇于拼搏的故事。

这是一套全面、系统地学习 ACM-ICPC 竞赛的知识类书籍；

这是一套详尽、深入地熟悉 ACM-ICPC 竞赛的算法及题目的手册类书籍；

这是一套程序设计、数据结构、算法等相关课程的拓展与提升类书籍；

这是一部上海交通大学 ACM-ICPC 队的成长史；

这是一部激励更多学子勇敢追寻并实现自己最初梦想的励志书。

历时 2 年零 5 个月，终于完成了本系列丛书，编者与队员有一种如释重负的感觉，因为我们把出版这套丛书看得很重，这是我们 16 年的经验与积累，希望对广大读者有用。

值此 ACM-ICPC 进入中国大陆 16 周年、上海交通大学获得 ACM-ICPC 世界冠军 10 周年之际，谨以此系列丛书——

纪念我们曾经走过的路、度过的岁月；

献给所有支持、帮助过我们的人……

俞 勇

2012 年 10 月于上海

前言

在 ACM 国际大学生程序设计竞赛（ACM International Collegiate Programming Contest, ACM-ICPC 或 ICPC）中，实现算法的能力是非常重要的。尤其是对新手来说，在了解到一个新的算法后，有时会对如何实现该算法产生困惑，也许并不能一下想到很好的实现，这时就需要参考一些已有的实现。

另外，ACM-ICPC 比赛中允许选手将一定量的（一般为 25 页）纸质资料带入比赛现场进行参考。队伍往往会将一些相对较难实现的常用算法的代码整理为 SCL（Standard Code Library，标准代码库）带入赛场，在需要的时候可以直接抄写已有代码，既节省时间，也保证了正确性。

因此我们出版这本收集了大量经典常用算法的用 C++ 语言实现的代码库，希望可以帮助读者学习算法以及准备比赛用的 SCL。

本书分为两个部分。第一部分为代码库，涵盖了大部分比赛常用的经典算法，包括数学、图论、数据结构、计算几何、论题选编五个大类，对每个算法的代码实现，都配有接口说明以及简略的算法阐述，便于读者理解。第二部分为贴士，收集了一些实用的知识点以及积分表，适合于带入赛场进行参考。

本书编写工作历时两年左右，参与编写工作的人员全部为上海交通大学 ACM-ICPC 队的现役队员。代码大多来自于往年上海交通大学 ACM-ICPC 队使用的 SCL 以及队员的日常训练。同时，本书的编写也得到上海交通大学 ACM-ICPC 队的退役队员大力帮助，他们参与了代码库的收集、整理、校验等工作。

参与本书写稿、审稿的人员主要有（按姓氏笔画为序）：尹天蛟，乌辰洋，任春旭，刘奇，刘彦，寿鹤鸣，李说，杨思逸，吴卓杰，张捷钧，陈明骋，陈泽佳，陈彬毅，陈爽，林承宇，金斌，郑翌，胡张广达，郭晓旭，曹雪智，康南茜，章雍哲，商静波，彭上夫，谭天，缪沛晗，瞿钧等。

在此，衷心感谢所有为此书出版做出直接或间接贡献的人！也真心祝愿此书能够在算法实现和 SCL 的准备上给读者带来帮助。

由于时间仓促，作者水平有限，疏漏、不当和不足之处在所难免，真诚地希望专家和读者朋友们不吝赐教。如果您能在阅读和使用此书过程中发现任何问题或有任何建议，恳请发邮件至 yyu@cs.stu.edu.cn，我们将不胜感激。

编 者

2012 年 10 月于上海

目 录

第一部分 算 法

第 1 章 数学	3
1.1 矩阵	3
1.1.1 矩阵类	3
1.1.2 Gauss 消元	4
1.1.3 矩阵的逆	6
1.1.4 常系数线性齐次递推	7
1.2 整除与剩余	9
1.2.1 欧几里得算法	9
1.2.2 扩展欧几里得	9
1.2.3 单变元模线性方程	10
1.2.4 中国剩余定理	11
1.2.5 求原根	13
1.2.6 平方剩余	14
1.2.7 离散对数	15
1.2.8 N 次剩余	16
1.3 素数与函数	18
1.3.1 素数筛法	18
1.3.2 素数判定	19
1.3.3 质因数分解	20
1.3.4 欧拉函数计算	21
1.3.5 Mobius 函数计算	23
1.4 数值计算	24
1.4.1 数值积分	24
1.4.2 高阶代数方程求根	26
1.5 其他	27
1.5.1 快速幂	27
1.5.2 进制转换	28
1.5.3 格雷码	29
1.5.4 高精度整数	30

1.5.5 快速傅立叶变换	35
1.5.6 分数类	37
1.5.7 全排列散列	38
第 2 章 图论	40
2.1 图的遍历及连通性	40
2.1.1 前向星	40
2.1.2 割点和桥	42
2.1.3 双连通分量	43
2.1.4 极大强连通分量 Tarjan 算法	45
2.1.5 拓扑排序	47
2.1.6 2SAT	49
2.2 路径	51
2.2.1 Dijkstra	51
2.2.2 SPFA	53
2.2.3 Floyd-Warshall	54
2.2.4 无环图最短路	55
2.2.5 第 k 短路	56
2.2.6 欧拉回路	59
2.2.7 混合图欧拉回路	61
2.3 匹配	64
2.3.1 匈牙利算法	64
2.3.2 Hopcroft-Karp 算法	66
2.3.3 KM 算法	68
2.3.4 一般图最大匹配	71
2.4 树	74
2.4.1 LCA	74
2.4.2 最小生成树 Prim 算法	77
2.4.3 最小生成树 Kruskal 算法	78
2.4.4 单度限制最小生成树	79

2.4.5	最小树形图	83	3.2.6	圆的面积并	144
2.4.6	最优比例生成树	85	3.3	三维计算几何	147
2.4.7	树的直径	87	3.3.1	三维点类	147
2.5	网络流	89	3.3.2	三维直线类	150
2.5.1	最大流 Dinic 算法	89	3.3.3	三维平面类	152
2.5.2	最小割	92	3.3.4	三维向量旋转	154
2.5.3	无向图最小割	93	3.3.5	长方体表面两点 最短距离	155
2.5.4	有上下界的网络流	95	3.3.6	四面体体积	156
2.5.5	费用流	97	3.3.7	最小球覆盖	158
2.6	其他	100	3.3.8	三维凸包	161
2.6.1	完美消除序列	100	3.4	其他	164
2.6.2	弦图判定	101	3.4.1	三角形的四心	164
2.6.3	最大团搜索算法	103	3.4.2	最近点对	166
2.6.4	极大团的计数	105	3.4.3	平面最小曼哈顿距离 生成树	167
2.6.5	图的同构	107	3.4.4	最大空凸包	171
2.6.6	树同构	108	3.4.5	平面划分	174
第 3 章	计算几何	112	第 4 章	数据结构	179
3.1	多边形	112	4.1	二叉堆	179
3.1.1	计算几何误差修正	112	4.2	并查集	183
3.1.2	计算几何点类	113	4.3	树状数组	184
3.1.3	计算几何线段类	115	4.4	左偏树	186
3.1.4	多边形类	117	4.5	Trie	188
3.1.5	多边形的重心	118	4.6	Treap	190
3.1.6	多边形内格点数	119	4.7	伸展树	193
3.1.7	凸多边形类	120	4.8	RMQ 线段树	199
3.1.8	凸多边形的直径	123	4.9	ST 表	201
3.1.9	半平面切割多边形	124	4.10	动态树	202
3.1.10	半平面交	126	4.11	块状链表	207
3.1.11	凸多边形交	128	4.12	树链剖分	210
3.1.12	多边形的核	129	第 5 章	论题选编	213
3.1.13	凸多边形与直线集交	130	5.1	字符串	213
3.2	圆	133	5.1.1	KMP	213
3.2.1	圆与线求交	133	5.1.2	扩展 KMP	214
3.2.2	圆与多边形交的面积	134	5.1.3	串的最小表示	216
3.2.3	最小圆覆盖	137			
3.2.4	圆与圆求交	138			
3.2.5	圆的离散化	140			

5.1.4	有限状态自动机	217
5.1.5	后缀数组	221
5.1.6	最长重复子串	223
5.1.7	最长公共子串	225
5.1.8	最长回文子串 manacher 算法	227
5.1.9	字符串散列	228
5.2	转换	229
5.2.1	星期计算	229
5.2.2	日期相隔天数计算	230
5.2.3	斐波那契进制转换	232
5.2.4	罗马进制转换	233
5.3	构造	235
5.3.1	幻方构造	235
5.3.2	N 皇后问题	237
5.3.3	旋转魔方	239
5.3.4	骑士周游问题	242
5.4	计算	245
5.4.1	表达式计算	245
5.4.2	最大权子矩形	247
5.4.3	矩形面积并	249
5.4.4	矩形并的周长	252
5.5	序列	255
5.5.1	第 k 小数	255
5.5.2	逆序对	256
5.5.3	最长公共子序列	257
5.5.4	最长公共上升子序列	259

第二部分 贴 士

第 6 章	代数	263
6.1	Bertrand 猜想	263

6.2	差分序列	263
6.3	威尔逊定理	263
6.4	约数个数	263
6.5	行列式的值	264
6.6	最小二乘法	264
第 7 章	解析几何	265
7.1	四边形	265
7.2	抛物线	265
7.3	双曲线	265
7.4	椭圆	266
第 8 章	平面立体几何	267
8.1	费马点	267
8.2	皮克定理	267
8.3	三角公式	267
8.4	三维几何体	268
8.5	托勒密定理	268
第 9 章	组合数学	269
9.1	Catalan 数	269
9.2	组合公式	269
第 10 章	图论	271
10.1	树的计数	271
10.2	有特殊条件的汉米尔顿回路	271
10.3	普吕弗序列	272
10.4	模 2 意义下的二分图匹配数	272
第 11 章	积分表	273

第一部分

算 法

1.1 矩 阵

1.1.1 矩阵类

【任务】

实现矩阵的基本变换。

【接口】

结构体: *Matrix*

成员变量:

`int n,m` 矩阵大小

`int a[][]` 矩阵内容

重载运算符: +、-、×

成员函数:

`void clear()` 清空矩阵

【代码】

```
1  const int MAXN=1010;
2  const int MAXM=1010;
3  struct Matrix{
4      int n,m;
5      int a[MAXN][MAXM];
6      void clear(){
7          n=m=0;
8          memset(a,0,sizeof(a));
9      }
10     Matrix operator +(const Matrix &b) const{
11         Matrix tmp;
12         tmp.n=n; tmp.m=m;
```

```

13         for (int i=0; i<n; ++i)
14             for (int j=0; j<m; ++j)
15                 tmp.a[i][j]=a[i][j]+b.a[i][j];
16         return tmp;
17     }
18     Matrix operator -(const Matrix &b) const{
19         Matrix tmp;
20         tmp.n=n; tmp.m=m;
21         for (int i=0; i<n; ++i)
22             for (int j=0; j<m; ++j)
23                 tmp.a[i][j]=a[i][j]-b.a[i][j];
24         return tmp;
25     }
26     Matrix operator *(const Matrix &b) const{
27         Matrix tmp;
28         tmp.clear();
29         tmp.n=n; tmp.m=b.m;
30         for (int i=0; i<n; ++i)
31             for (int j=0; j<b.m; ++j)
32                 for (int k=0; k<m; ++k)
33                     tmp.a[i][j]+=a[i][k]*b.a[k][j];
34         return tmp;
35     }
36 };

```

【使用范例】

参见程序 POJ3420.CPP。

1.1.2 Gauss 消元

【任务】

给一个 n 元一次方程组，求它们的解集。

【说明】

将方程组做成矩阵形式，再利用三种初等矩阵变换，得到上三角矩阵，最后回代得到解集。

【接口】

```
int solve(double a[][MAXN], bool l[], double ans[], const int& n);
```

复杂度: $O(n^3)$

输入: a 方程组对应的矩阵
 n 未知数个数
 l, ans 存储解, $l[]$ 表示是否为自由元
输出: 解空间的维数

【代码】

```
1  inline int solve(double a[][MAXN], bool l[], double ans[],
2  const int& n) {
3      int res = 0, r = 0;
4      for (int i = 0; i < n; ++i)
5          l[i] = false;
6      for (int i = 0; i < n; ++i) {
7          for (int j = r; j < n; ++j)
8              if (fabs(a[j][i]) > EPS) {
9                  for (int k = i; k <= n; ++k)
10                     swap(a[j][k], a[r][k]);
11                     break;
12             }
13             if (fabs(a[r][i]) < EPS) {
14                 ++res;
15                 continue;
16             }
17             for (int j = 0; j < n; ++j)
18                 if (j != r && fabs(a[j][i]) > EPS) {
19                     double tmp = a[j][i] / a[r][i];
20                     for (int k = i; k <= n; ++k)
21                         a[j][k] -= tmp * a[r][k];
22                 }
23             l[i] = true, ++r;
24         }
25         for (int i = 0; i < n; ++i)
26             if (l[i])
27                 for (int j = 0; j < n; ++j)
28                     if (fabs(a[j][i]) > 0)
29                         ans[i] = a[j][n] / a[j][i];
30         return res;
31     }
```

【使用范例】

参见程序 POJ1830.CPP。

1.1.3 矩阵的逆**【任务】**

给一个矩阵，求它的逆。

【说明】

将原矩阵 A 和一个单位矩阵 E 作成大矩阵 (A, E) ，用初等行变换将大矩阵中的 A 变为 E ，则会得到 (E, A^{-1}) 的形式。

【接口】

`void inverse(vector<double> A[], vector<double> C[], int N);`

复杂度： $O(n^3)$

输 入： A 原矩阵

C 逆矩阵

N 矩阵的阶数

【代码】

```

1  inline vector<double> operator * (vector<double> a, double b) {
2      int N = a.size();
3      vector<double> res(N, 0);
4      for (int i = 0; i < N; ++i)
5          res[i] = a[i] * b;
6      return res;
7  }
8  inline vector<double> operator - (vector<double> a, vector<double> b) {
9      int N = a.size();
10     vector<double> res(N, 0);
11     for (int i = 0; i < N; ++i)
12         res[i] = a[i] - b[i];
13     return res;
14 }
15 inline void inverse(vector<double> A[], vector<double> C[], int N) {
16     for (int i = 0; i < N; ++i)
17         C[i] = vector<double>(N, 0);

```

```

18     for (int i = 0; i < N; ++i)
19         C[i][i] = 1;
20     for (int i = 0; i < N; ++i) {
21         for (int j = i; j < N; ++j)
22             if (fabs(A[j][i]) > 0) {
23                 swap(A[i], A[j]);
24                 swap(C[i], C[j]);
25                 break;
26             }
27         C[i] = C[i] * (1 / A[i][i]);
28         A[i] = A[i] * (1 / A[i][i]);
29         for (int j = 0; j < N; ++j)
30             if (j != i && fabs(A[j][i] > 0)) {
31                 C[j] = C[j] - C[i] * A[j][i];
32                 A[j] = A[j] - A[i] * A[j][i];
33             }
34     }

```

【使用范例】

参见程序 POJ1166.CPP。

1.1.4 常系数线性齐次递推

【任务】

已知 $f_x = a_0 f_{x-1} + a_1 f_{x-2} + \cdots + a_{n-1} f_{x-n}$ 和 $f_0, f_1, \cdots, f_{n-1}$, 给定 t , 求 f_t 。

【说明】

f 的递推可以看成是一个 $n \times n$ 的矩阵 A 乘以一个 n 维列向量 β , 因为矩阵乘法满足结合律, 用快速幂可以加速。其中,

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ a_{n-1} & a_{n-2} & a_{n-3} & \cdots & a_0 \end{bmatrix}, \quad \beta = \begin{bmatrix} f_{x-n} \\ f_{x-n+1} \\ \vdots \\ f_{x-2} \\ f_{x-1} \end{bmatrix}$$

【接口】

```
int solve(int a[], int b[], int n, int t);
```

复杂度: $O(n^3 \log t)$