工作流可满足性(≠,=)计数及其#P完全性

翟治年[1],卢亚辉[2],余法红[3],周武杰[1],向 坚[1],吴茗蔚[1]
(1. 浙江科技学院信息与电子工程学院,浙江杭州 310023; 2. 深圳大学计算机学院,广东深圳 518060;
3. 嘉兴学院数理与信息学院,浙江嘉兴 314001)

摘 要: 工作流可满足性(WS)是资源分配对访问控制(AC)策略提出的基本要求.相关工作主要围绕WS决策问题展开,通过找到一个具体的解来说明AC策略的正确性.然而为了进一步验证AC策略在资源异常情况下的合理性,统计所有解的数量将更有帮助.本文对互斥和绑定约束下的WS计数问题进行研究,通过构造从典范性#P完全问题#3SAT到该问题的多项式计数归约,证明其属于#P完全问题,为其恰当地求解奠定了理论基础.

关键词: 工作流;访问控制;授权;约束;资源分配;可满足性
中图分类号: TP309 **文献标识码:** A **文章编号:** 0372-2112 (2017)03-0605-07
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2017.03.015

Counting Workflow Satisfiability(≠,=) and Its #P Completeness

ZHAI Zhi-nian[1], LU Ya-hui[2], YU Fa-hong[3], ZHOU Wu-jie[1], XIANG Jian[1], WU Ming-wei[1]
(1. *School of Information and Electronic Engineering, Zhejiang University of Science and Technology, Hangzhou, Zhejiang* 310023, *China*;
2. *School of Computer, Shenzhen University, Shenzhen, Guangdong* 518060, *China*;
3. *College of Mathematics Physics and Information, Jiaxing University, Jiaxing, Zhejiang* 314001, *China*)

Abstract: Workflow satisfiability (WS) is an essential claim to access control (AC) policies from the view of resource allocation. So far, the related researches are concentrated on the decision problem of WS, which finds a single solution to show the correctness of an AC policy. However, to further verify its rationality under resource exception, and to count all the solutions will be more useful. In this paper, the counting problem of WS with exclusion and binding constraints is addressed. The problem is proved to be #P complete by constructing a polynomial time counting reduction from the well-known #P complete problem of #3SAT to it, and then gets a theoretical basis to be solved appropriately.

Key words: workflow; access control; authorization; constraint; resource allocation; satisfiability

1 引言

工作流系统需要组织资源来执行一组指向特定目标的业务步骤.其关键环节是资源分配,它在工作流每次运行(称为案例)时,为每个步骤指派唯一的执行用户[1,2].为防止非法和不当访问,需通过一定的访问控制(Access Control, AC)策略对资源分配进行制约.AC策略主要包括授权和约束.其中授权将每个步骤的候选执行资格分配给一组用户.而约束作用于利益相关的步骤,要求其执行用户之间满足某种关系.

这就引出了一个有基础意义的问题:是否存在恰当的资源分配,使任何步骤均由其授权用户执行,且不违反彼此间的约束[3].此问题称为工作流可满足性(Workflow Satisfiability, WS).它关系到工作流的可行性,反映了AC策略规划是否正确.因为步骤的执行需要时间和经济成本,WS不能靠执行期发现问题随时调整的方式来保证,而须在AC规划阶段加以验证.

自1999年文献[4]涉及相关问题以来,WS研究已经取得了一些成果[1,5~8].这些工作多以找到WS的一个解为目标(文献[4]枚举所有解),由此说明WS成立.然而,历史更久的合取范式(Conjunctive Normal Form, CNF)可满足性(Satisfiability, SAT)[9]和约束可满足性(Constraint Satisfiability, CS)等经典问题,均包括决策(求一个具体的解)和计数(统计所有解的数量)两个

收稿日期:2015-04-20;修回日期:2015-07-08;责任编辑:梅志强
基金项目:国家自然科学基金(No. 61502429, No. 61302112);浙江省自然科学基金(No. LQ15F020010, No. LY16F020027);浙江省教育厅科研项目(No. Y201533771);钱江人才计划(No. QJD1402023)

方面,以不同方式说明 WS 是否成立. SAT 和 CS 计数研究已经取得了丰富成果[10~13],而现有 WS 研究均未考虑计数问题,是一个显著的缺失. 实际上,任何 NP 问题 q 都有其计数问题,记为#q,它统计 q 所有解的数量. 计数给出了 NP 问题解集的关键特征,具有基本的理论意义,在人工智能、可靠性等等领域有重要应用[14]. 具体到#WS 计数,不仅可用于验证 AC 策略的正确性,而且有助于其合理规划,举例如下:

例 1 一个借款工作流 W 的控制流如图 1 所示. 它包括借款(s_1)、审核(s_2)、付款(s_3)和报销(s_4)等 4 个步骤. 在 s_1 中申请从某项目经费借款进行一项支付;在 s_2 中对借款申请进行审核;在 s_3 中按借款金额转账给收款方;在 s_4 中凭发票从项目中报销冲抵. 图 2 利用互斥($\neq$)和绑定($=$)两种约束描述了如下业务规则:为避免借款人自审核,并进一步保证借款审核与后续报销相独立,s_1 和 s_2、s_2 和 s_4 均不能由同一人执行;为分离单据审核与实际付款职能,s_2 和 s_3 不能由同一人执行;而为了保证借款及其冲抵过程的完整性,s_1 和 s_4 必须由同一人执行.

设有 3 个用户 u_1、u_2、u_3 负责此工作流的案例处理. 基于图 2 确定的约束关系,图 3(1)给出了一种可能的授权(每个步骤旁列出其所有授权用户).

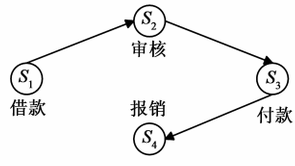


图1 借款工作流W

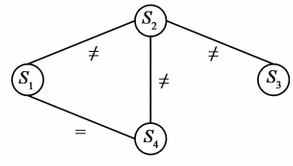


图2 W的约束关系

由此形成的 AC 策略只有一种可行的资源分配,可用函数 π 表示为

$$\pi(s_1)=\pi(s_4)=u_1,\pi(s_2)=u_2,\pi(s_3)=u_3$$

它依赖于每一个用户,无论谁脱离岗位,都会造成新的案例资源分配无解. 该 AC 策略使工作流决策可满足,但却对资源异常缺乏必要的鲁棒性. 通过 AC 策略特别是授权部分的合理规划,可有效改善这种状况.

将策略 1 中 s_2、s_4 的授权用户 u_2、u_3 对调,得到如图 3(2)所示的策略 2. 它有 4 种可行资源分配:

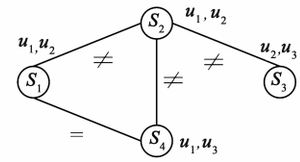
$$A_1:\pi(s_1)=\pi(s_4)=u_1,\pi(s_2)=u_3,\pi(s_3)=u_2;$$

$$A_2:\pi(s_1)=\pi(s_4)=u_2,\pi(s_2)=u_3,\pi(s_3)=u_2;$$

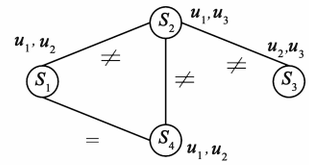
$$A_3:\pi(s_1)=\pi(s_4)=u_2,\pi(s_2)=u_1,\pi(s_3)=u_2;$$

$$A_4:\pi(s_1)=\pi(s_4)=u_2,\pi(s_2)=u_1,\pi(s_3)=u_3,$$

其中 A_2 不依赖于 u_1,A_3 不依赖 u_3. 当缺少用户 u_1 或 u_3 时,都不会导致案例处理无解. 该策略同样决策可满足,但是比策略 1 更为合理,更能抵抗资源异常影响. 所谓合理性,其精确度量极为复杂,但它必然与相应资源分配解集的统计特性有关,其中最基本的指标是解集大小. 在其它特征相近时,解集越大,解的多样性越好. 当个别资源异常时,替代的资源分配方式越多,不受干扰的可能性也越大.



(a)策略1



(b)策略2

图3 W的两种AC策略

综上,只研究 WS 决策并不能满足 AC 策略规划的需要. 统计某种策略下资源分配解的个数,即 WS 计数,将为合理的 AC 规划提供更重要的参考指标. WS 决策的解可在多项式时间内验证,因此属于 NP 问题. 所有 NP 问题的计数问题构成一个复杂性类,称为#P[10]. 人们已经发现了类似于 NP 完全性的#P 完全性[11]. 所有#P完全问题可以多项式时间相互计数归约,若其中一个找到多项式时间算法,则其它所有问题均可以多项式时间求解. 判断#P 完全性通常是计数问题研究的第一步. 由此可确定问题是否不太可能找到多项式时间算法,避免无谓的尝试. 进而,#P 完全性在表面上不同的问题之间建立了联系,从而有可能借助其它领域的成果解决当前问题.

在现有 WS 研究中,主要是文献[3]完成了 WS($\neq$)决策的复杂性归类,证明其是 NP 完全的. 其方法是将 NP 完全的图着色问题多项式归约为 WS($\neq$). 由于图着色其实就是 WS($\neq$)的图形表示,这一证明是比较直接的. 然而,#WS 的复杂性归类尚未见研究. 具体到基本的#WS($\neq$),由于主要的#P 完全问题[10~14]中未见与其结构相近者,证明其#P 完全性将更为困难.

本文对#WS($\neq$)的推广形式,即考虑互斥和绑定两种约束的#WS($\neq$, $=$),进行研究,主要贡献是证明了问题的#P 完全性,为其应用求解奠定了理论基础.

2　#WS(≠,=)问题定义

令 $S=\{s_1,s_2,\cdots,s_{|S|}\}$ 和 $U=\{u_1,u_2,\cdots,u_{|U|}\}$ 分别表示步骤集和用户集,首先给出一些基本定义.

定义 1　工作流 $W=\langle S,\leqslant\rangle$ 是步骤的偏序集,其中 $\leqslant$ 为 S 上的偏序关系,表示步骤间的执行顺序. 任取 s, $s'\in S$,若 $s<s'$,则 s 先于 s' 执行. 将工作流的一次执行称为一个案例.

定义 2　AC 策略是一个三元组 $\langle A,X,B\rangle$,其中:

(1) $A\subseteq S\times U$ 为授权,表示以 S 为步骤集的工作流在用户集 U 上的权限分配情况. 若 $\langle s,u\rangle\in A$,则用户 u 是步骤 s 的一个候选执行者,称 u 是 s 的授权用户, s 是 u 的授权步骤;

(2) $X\subseteq S\&S$ 为互斥约束,表示步骤间的职责分离关系. 若 $(s,s')\in X$,则在同一案例中, s 和 s' 不能由同一用户执行. 互斥禁止某些利益冲突的步骤(例如出款和记账、报销和审核等等(由同一用户执行,以免借机欺诈. 不引起混淆时,可将 X 简记为 $\neq$;

(3) $B\subseteq S\&S$ 为绑定约束,表示步骤间的职责绑定关系. 若 $(s,s')\in B$,则在同一案例中, s 和 s' 必须由同一用户执行. 绑定要求利益相关的两个步骤由同一用户执行. 例如材料采购流程,验货必须由采购申请人来完成. 不引起混淆时,可将 B 简记为 =.

定义 2　资源分配是一个函数 $\pi:S\to U$,为每个步骤指派实际的执行用户. 同一工作流的每个案例都需要单独进行资源分配.

此时可以给出 WS(≠,=)及其计数问题的定义.

定义 3　一个互斥和绑定约束下的 WS 问题是一个 AC 策略,它的解是满足如下条件的资源分配 π:

(1)任取 $s\in S$, $\langle s,\pi(s)\rangle\in A$.

(2)任取 $(s,s')\in X$, $\pi(s)\neq\pi(s')$.

(3)任取 $(s,s')\in B$, $\pi(s)=\pi(s')$.

将所有此类问题的集合记为 WS(≠,=). $\langle A,X,B\rangle\in$ WS(≠,=)的解也称为 $\langle A,X,B\rangle$ 的可满足解. 将所有解的个数记为 $\#\langle A,X,B\rangle$,相应的计数问题仍用 $\langle A,X,B\rangle$ 表示,问题的集合记为#WS(≠,=). $\langle A,X,B\rangle\in$#WS(≠,=)的解称为 $\langle A,X,B\rangle$ 的可满足计数解.

3　#WS(≠,=)的#P 完全性

现在考虑#WS(≠,=)的复杂性归类. 已知 NP 是非确定型图灵机多项式时间内可计算或(确定型图灵机)多项式时间可验证的问题,而#P 是 NP 的计数问题. 由于 WS(≠,=)解的正确性可在多项式时间内验证(首先逐一检查每个步骤的指派用户是否符合授权,然后逐一检查每条约束是否得到满足),故 WS(≠,=)属于 NP,而#WS(≠,=)属于#P. 进一步,如果 A 是一个#P 问题,且任何#P 问题 B 都可在多项式时间内归约为 A,并且存在函数 $\varphi:N\to N$,将 A 的计数映射为 B 的计数(不一定相等),则称 A 是#P 完全的,而这一归约过程称为多项式计数归约[10]. 为证明 A 的#P 完全性,只需证明一个已知的#P 完全问题 B 可以多项式计数归约为 A. #SAT 是典范的#P 完全问题[11,10],其特殊形式#3SAT 也较早被证明是#P 完全的[10]. 本文将利用#3SAT 来证明#WS(≠,=)的#P 完全性.

定理 1　**#WS(≠,=)问题是#P 完全的**

证明　如前述#WS(≠,=)本身已经是#P 问题. 为证明任何#P 问题可多项式计数归约为#WS(≠,=),可将#3SAT 多项式计数归约为#WS(≠,=).

任取 3CNF(每个子句至多包含 3 个不同变量的文字的 CNF) f,设其包含 m 个变量 $v_1,v_2,\cdots v_m$, n 个子句 $D_1,D_2,\cdots D_n$. 下面先对 f 的成分和结构进行翻译,构造出一个#WS(≠,=)问题 $\langle A,X,B\rangle$,使得 f 可满足当且仅当 $\langle A,X,B\rangle$ 有可行资源分配,然后分析 f 的成真赋值数与 $\langle A,XM,B\rangle$ 的可行资源分配数的对应关系.

下面给出 $\langle A,X,B\rangle$ 的构造,并解释其与 f 的关系. $\langle A,X,B\rangle=\langle S\times U,X,\varnothing\rangle$,而 U,S 和 X 构造为:

(1) $U=\{u_0,u_1,u_2,\cdots u_m\}$,共 $m+1$ 个用户.

(2) $S=V\cup C\cup U\cup P$,其中 V 是变量文字步骤集.

$v_i(1\leqslant i\leqslant m)$ 和 $\bar{v}_i$ 是变量 v_i 产生的两个文字,根据它们各自构造一个步骤. 每个步骤表示该变量产生的一个文字,两个步骤合起来,可以表示这一变量. 变量文字步骤集是所有可能的标准文字构件,覆盖了所有子句中的所有文字.

$C=\bigcup_{j=1}^{n}C_j$, $C_j=\{x_i\mid x_i\lhd D_j\}$,为子句文字步骤集. 其中 x_i 表示变量 v_i 产生的文字(可能是 v_i 或 $\bar{v}_i$), $x_i\lhd D_j$ 表示 x_i 在子句 D_j 中出现. 这样,子句 D_j 中出现的每个文字将对应于 C_j 中的一个步骤,而 C_j 中的所有步骤共同表示子句 D_j. 如果不同的 D_j 中出现同一个文字 x_i,在它们相应的 C_j 中将对应着标记相同(均为 x_i)的不同步骤,通常根据上下文来区别. 类似地, C 和 V 之间也存在标记相同的不同步骤. 可通过加注所属集合来区分,例如 $x_i^{(C_j)}$ 表示 C_j 中的 x_i.

S 中还包含着 U,也就是对每个用户,都会构造一个相应的步骤,称为用户步骤. 如前所述,用 $\langle A,X,B\rangle$ 来表示一个 3CNF 之后,需要用资源分配来表示赋值. 由于变量用一对文字来表示,关键在于文字的值如何表示. 既然文字用步骤来表示,它的值应该用步骤的执行用户来表示. 然而文字非真即假,步骤的执行用户却可能有 $m+1$ 个. 为消除这种差别,根据每个用户构造一个用户步骤,在(子句或变量)文字步骤 x_i 与所有非 u_i、u_0 的用户步骤之间添加互斥约束(见后面 X 的构造),

从而文字步骤 x_i 只能被指派给用户 u_i 或 u_0. 不妨约定：指派给 u_0 表示文字 x_i 为真，指派给 u_i 表示该文字为假. 这样，文字赋值为真只有一种表示，而赋值为假对不同变量的文字有不同的表示. 至此就解释了为什么 S 中也包含 U.

$P=\{p_{t_1,t_2}^{(j)}\mid C_j=\{x_{t_1},x_{t_2},x_{t_3}\},1\leqslant t_1<t_2<t_3\leqslant m\}$ 是为包含三个文字的子句 D_j 额外构造的枢纽步骤. 设 C_j 包含的三个子句文字步骤为 x_{t_1},x_{t_2},x_{t_3}，取下标较小的两个来构造子句 D_j 的枢纽步骤，记为 $p_{t_1,t_2}^{(j)}$. 后面将在 X 中构造相关约束，利用 $p_{t_1,t_2}^{(j)}$ 表示出子句文字 x_{t_1},x_{t_2},x_{t_3} 之间的析取关系.

接下来给出 $X\subseteq S\&S$ 的构造，它包括如下子集：

(3) $X=X_{UU}\cup X_{VV}\cup X_{VU}\cup X_{CV}\cup X_{CU}\cup X_{PU}\cup L$，其中 $X_{UU}=\{(u_s,u_t)\mid 0\leqslant s\neq t\leqslant m\}$，是用户步骤两两之间的互斥约束. 在进行资源分配时，要为每个步骤指派一个执行用户. 而用户步骤的特殊性在于，其执行用户就是该步骤所表示的用户（在 S 为何包含 U 的解释中，已经隐含用到了这一点）. X_{UU} 中的互斥约束可保证不同的用户步骤，其执行用户不同. 然而在此约束下进行资源分配，仍有多达 $(m+1)!$ 种方式，用户步骤 u_s 不一定被指派给用户 u_s. 不过改变排列顺序导致的不同资源分配没有实质差异，不妨约定指派给用户步骤 u_s 的用户就是 u_s，若非用户步骤与用户步骤 u_s 指派了相同的用户，则认为此步骤的执行用户是 u_s.

$X_{VV}=\{(v_i,\bar{v}_i)\mid i=1,2,\cdots,m\}$，是（同一变量）相反文字步骤之间的互斥. 因为文字的真值相反，所以相应步骤的执行用户不能相同.

$X_{VU}=\{(x_s,u_t)\mid s\neq t,t\neq 0,x_s\in V\}$，是每个变量文字步骤 x_s 与除 u_s、u_0 之外所有用户步骤之间的互斥，用来保证 x_s 的执行用户只能为 u_s 或 u_0. 这反映了变量文字 x_s 的取值非真即假.

$X_{CV}=\bigcup_{j=1}^{n}X_{C_jV}$，是子句与变量的文字步骤间的互斥，其中 $X_{C_jV}=\bigcup_{j=1}^{m_j}X_{C_{js_i}V}(1\leqslant m_j\leqslant 3)$ 是与子句 D_j 有关的互斥，式中 m_j 为子句 D_j 所含文字的数目，$X_{C_{js_i}V}=\{(x_{s_i},\bar{x}_{s_i},)\mid x_{s_i}\in C_j,\bar{x}_{s_i}\in V\}$ 是与 D_j 中第 i 个文字（其变量是 v_{s_i}）有关的互斥. 由 $X_{C_{js_i}V}$ 的构造可知，C_j 中的每个文字步骤与其在 V 中相反文字的步骤之间是互斥的（例如 C_j 中的步骤 $x_{s_i}=\bar{v}_{s_i}$，则其与 V 中的步骤 $\bar{x}_{s_i}=\bar{\bar{v}}_{s_i}=v_{s_i}$ 互斥），从而必须由不同用户执行. 这反映了相反的子句文字与变量文字的真值不同.

$X_{CU}=\bigcup_{j=1}^{n}X_{C_jU}$，是子句文字步骤与用户步骤间的互斥，其中跟子句 D_j 有关的互斥 $X_{C_jU}=\bigcup_{i=1}^{m_j}X_{C_{js_i}U}(1\leqslant m_j\leqslant 3)$. 而 $X_{C_{js_i}U}=\{(x_{s_i},u_t)\mid t\neq s_i,t\neq 0,x_{s_i}\in C_j\}$，表示 C_j 中的步骤 x_{s_i} 与 u_0,u_{s_i} 之外的每个用户步骤是互斥的. X_{C_jU} 中的约束用来保证 C_j 中每个基于变量 v_{s_i} 的文字步骤只能由用户 u_0 或 u_{s_i} 执行. 这反映了子句文字是非真即假的.

X_{VV}、X_{C_jV} 和 X_{C_jU} 共同保证 C_j 中的任何步骤与 V 中相同标记的步骤由相同用户执行. 这反映了同一赋值下的同一文字，不论是否在子句中出现，其值唯一.

现在，每个文字步骤只能由两个候选用户之一来执行，反映了文字非真即假. 接下来，将在每个 C_j 的文字步骤之间建立析取关系，使它们的执行用户中至少有一个是 u_0，以反映子句 D_j 的文字中至少有一个为真. 这通过 X_{PU} 和 L 中的约束来实现.

$X_{PU}=\{(p_{t_1,t_2}^{(j)},u_i)\mid p_{t_1,t_2}^{(j)}\in P,i\notin\{t_1,t_2,0\}\}$，是三文字子句枢纽步骤与用户步骤之间的互斥，可以保证三文字子句的枢纽步骤 $p_{t_1,t_2}^{(j)}$ 只能由用户 u_{t_1}、u_{t_2} 或 u_0 来执行，稍后将解释其用途所在.

$L=L_1\cup L_2\cup L_3$，是同一子句各文字步骤间的互斥，用来表示子句内部析取关系. 其中，$L_1=\{(x_s,u_s)\mid C_j=\{x_s\}\}$，是单文字子句 D_j 的文字步骤 x_s 与对应的用户步骤 u_s 之间的互斥. 前面已经有约束保证步骤 x_s 只能由用户 u_0 或 u_s 执行，L_1 中的互斥进一步保证步骤 x_s 只能由用户 u_0 执行. 相关构造反映的要求是：若一个子句只包含一个文字，该文字必须为真. 此时不难证明：D_j 的成真赋值的数目 $=C_j$ 的可行资源分配的数目 $=1$.

$L_2=\{(\bar{x}_{t_1},\bar{x}_{t_2})\mid\bar{x}_{t_1},\bar{x}_{t_2}\in V,C_j=\{x_{t_1},x_{t_2}\}\}$，是双文字子句 D_j 对应的两个文字步骤 x_{t_1},x_{t_2} 各自在 V 中的反文字步骤 $\bar{x}_{t_1},\bar{x}_{t_2}$ 之间的互斥. 它保证变量文字步骤 $\bar{x}_{t_1},\bar{x}_{t_2}$ 不能由同一用户执行（只有一种可能，即 $\bar{x}_{t_1},\bar{x}_{t_2}$ 均由 u_0 执行），也就是 $\bar{x}_{t_1},\bar{x}_{t_2}$ 必有一个不由 u_0 执行，再由 X_{CV} 的约束，子句文字步骤 x_{t_1},x_{t_2} 必有一个由用户 u_0 执行. 相关构造反映的要求是：若一个子句恰好包含两个文字，其中必有一个为真. 不难证明：D_j 的成真赋值的数目 $=C_j$ 的可行资源分配的数目 $=3$.

$L_3=\bigcup_{C_j=\{x_{t_1},x_{t_2},x_{t_3}\}}\{(p_{t_1,t_2}^{(j)},x_{t_1}),(p_{t_1,t_2}^{(j)},x_{t_2}),(p_{t_1,t_2}^{(j)},\bar{x}_{t_3})\mid p_{t_1,t_2}^{(j)}\in P,1\leqslant t_1<t_2<t_3\leqslant m,\bar{x}_{t_3}\in V\}$，是三文字子句 D_j 前两个文字相应步骤 x_{t_1},x_{t_2}，剩余文字 x_{t_3} 在 V 中的反文字步骤 $\bar{x}_{t_3}$、以及该子句的枢纽步骤 $p_{t_1,t_2}^{(j)}$ 之间的互斥. 它所保证的目标是：子句文字步骤 x_{t_1},x_{t_2},x_{t_3} 中必有一个由 u_0 执行. 当 x_{t_1},x_{t_2} 中有一个由 u_0 执行时，目标自动实现. 否则，当 x_{t_1},x_{t_2} 的执行用户都不是 u_0 时（一个为 u_{t_1}，一个为 u_{t_2}），$(p_{t_1,t_2}^{(j)},x_{t_1})$，$(p_{t_1,t_2}^{(j)},x_{t_2})$ 使得 $p_{t_1,t_2}^{(j)}$ 只能由用户 u_0 执行. 此时，$(p_{t_1,t_2}^{(j)},\bar{x}_{t_3})$ 使 V 中的变量文字步骤 $\bar{x}_{t_3}$ 不能由 u_0 执行，从而子句文字步骤 x_{t_3} 必须由 u_0 执行（根据 X_{C_jV} 中的约束），前述目标同样得到保证. 相关构造反映的要求是：若一个子句包含三个文字，其中必须有一个取真值. 不难证明：D_j 的成真赋值的数目 $=C_j$ 的可行资源分

配的数目 =7.

由上述构造及其意图解释不难看出,f 可满足当且仅当 C 有可行资源分配. 接下来进一步分析 f 的赋值与整个 S 的资源分配的数量对应关系.

给定 f 的赋值,由于 X_{UU} 的构造使得 U 中的用户步骤两两互斥,需要为它们指派不同的执行用户,这相当于将所有用户进行全排列,共有 $(m+1)!$ 类方案,其本质是相同的,我们只考虑其中一类,也就是将用户步骤 $u_i(0\leqslant i\leqslant m)$ 指派给用户 u_i 执行的方案.

在此基础上,考虑 C 中步骤的资源分配. 由于 X_{CV} 和 X_{CU} 等构造,C 中的子句文字步骤 x_i 只能由用户 u_0 或 u_i 执行. 且由于 L 的构造,D_j 的成真赋值的数目 $=C_j$ 的可行资源分配的数目. 进而,f 的成真赋值数 = 每个 D_j 的成真赋值数的乘积 = 每个 C_j 的可行资源分配数的乘积 = C 的可行资源分配数. 不过,整个 S 的可行资源分配数尚未就此确定.

首先,若存在三文字子句,则有相应的枢纽步骤,需要考虑是否影响可行资源分配的数目.

由 X_{PU} 和 L_3 的构造可知,子句 $x_{t_1}\vee x_{t_2}\vee x_{t_3}$ 的枢纽步骤 $p^{(j)}_{t_1,t_2}$ 只能由用户 u_{t_1}、u_{t_2} 或 u_0 来执行,且其执行用户与 x_{t_1}、x_{t_2} 和 $\bar{x}_{t_3}$ 的都不同. 分为如下 3 种情况:

(1)当子句文字步骤 x_{t_1},x_{t_2} 均由 u_0 执行时,$p^{(j)}_{t_1,t_2}$ 的执行用户可以是 u_{t_1} 或 u_{t_2},两种方案并无本质差别,不妨取其执行用户为 u_{t_1}.

(2)当子句文字步骤 x_{t_1} 由 u_0 执行而 x_{t_2} 由 u_{t_2} 执行时,$p^{(j)}_{t_1,t_2}$ 的执行用户只能是 u_{t_1}.

(3)当子句文字步骤 x_{t_1} 由 u_{t_1} 执行而 x_{t_2} 由 u_0 执行,$p^{(j)}_{t_1,t_2}$ 的执行用户只能是 u_{t_2}.

由上述分析,可以认为给定 C 的可行资源分配后,各枢纽步骤的资源分配也随之唯一确定.

其次,V 中的步骤尚未指派执行用户,需要考虑其是否影响可行资源分配的数目. 对于给定的 f,V 中每个步骤对应的变量必定在 f 的子句中以文字形式出现过一次,否则不会被列入变量集合. 因此给定 C 的可行资源分配后,根据 X_{CV}、X_{VV} 以及 X_{VU} 等约束,V 中每个步骤的资源分配必然唯一确定.

此时,可以认为 S 的可行资源分配数 $=C$ 的可行资源分配数 $=f$ 的成真赋值数,$\langle A,X,B\rangle\in$ #WS(≠,=) 的可行资源分配数等于 $f\in$ 3CNF 的成真赋值数. 这样就得到了一种从#3SAT 到#WS(≠,=)的计数归约方式. 再由 $\langle A,X,B\rangle$ 的构造过程可知,这一归约可在关于 m 和 n 的多项式时间内完成. 因此,#3SAT 可在多项式时间内计数归约为#WS(≠,=),定理得证.

例 2　3CNF 公式 $f=(\bar{v}_1\vee v_2\vee\bar{v}_3)\wedge(v_1\vee v_3)\wedge v_2$ 的变量数和子句数均为 3. 它有如下 3 个成真赋值:①$v_1=1,v_2=1,v_3=0$;② $v_1=0,v_2=1,v_3=1$;③$v_1=1,v_2=1,v_3=1$

构造#WS(≠,=)问题 $\langle A,X,B\rangle=\langle S\times U,X,\varnothing\rangle$,其中 $U=\{u_0,u_1,u_2,u_3\}$,而 S 和 X 分别为:

(1)$S=V\cup C\cup U\cup P$,

$V=\{v_1,v_2,v_3\}\cup\{\bar{v}_1,\bar{v}_2,\bar{v}_3\}$,

$C=\bigcup_{j=1}^{3}C_j$,$C_1=\{\bar{v}_1,v_2,\bar{v}_3\}$,$C_2=\{v_1,v_3\}$,$C_3=\{v_2\}$,

$P=\{p^{(1)}_{1,2}\}$,$p^{(1)}_{1,2}$ 是 $D_1=\bar{v}_1\vee v_2\vee\bar{v}_3$ 的枢纽步骤.

(2)$X=X_{UU}\cup X_{VV}\cup X_{VU}\cup X_{CV}\cup X_{CU}\cup X_{PU}\cup L$,

$X_{UU}=\{(u_0,u_1),(u_0,u_2),(u_0,u_3),(u_1,u_2),(u_1,u_3),(u_2,u_3)\}$,

$X_{VV}=\{(v_1,\bar{v}_1),(v_2,\bar{v}_2),(v_3,\bar{v}_3)\}$,

$X_{VU}=\{(v_1,u_2),(v_1,u_3),(\bar{v}_1,u_2),(\bar{v}_1,u_3),(v_2,u_1),(v_2,u_3),(\bar{v}_2,u_1),(\bar{v}_2,u_3),(v_3,u_1),(v_3,u_2),(\bar{v}_3,u_1),(\bar{v}_3,u_2)\}$,

$X_{CV}=\bigcup_{j=1}^{3}X_{C_jV}$,

$X_{C_1V}=X_{C_{11}V}\cup X_{C_{12}V}\cup X_{C_{13}V}$,

$X_{C_{11}V}=\{(\bar{v}_1^{(C_1)},v_1^{(V)})\}$,

$X_{C_{12}V}=\{(v_2^{(C_1)},\bar{v}_2^{(V)})\}$,

$X_{C_{13}V}=\{(\bar{v}_3^{(C_1)},v_3^{(V)})\}$,

$X_{C_2V}=X_{C_{21}V}\cup X_{C_{23}V}$,

$X_{C_{21}V}=\{(v_1^{(C_2)},\bar{v}_1^{(V)})\}$,

$X_{C_{23}V}=\{(v_3^{(C_2)},\bar{v}_3^{(V)})\}$,

$X_{C_3V}=X_{C_{32}V}=\{(v_2^{(C_3)},\bar{v}_2^{(V)})\}$,

$X_{CU}=\bigcup_{j=1}^{3}X_{C_jU}$,

$X_{C_1U}=X_{C_{11}U}\cup X_{C_{12}U}\cup X_{C_{13}U}$,

$X_{C_{11}U}=\{(\bar{v}_1^{(C_1)},u_2),(\bar{v}_1^{(C_1)},u_3)\}$,

$X_{C_{12}U}=\{(v_2^{(C_1)},u_1),(v_2^{(C_1)},u_3)\}$,

$X_{C_{13}U}=\{(\bar{v}_3^{(C_1)},u_1),(\bar{v}_3^{(C_1)},u_2)\}$,

$X_{C_2U}=X_{C_{21}U}\cup X_{C_{23}U}$,

$X_{C_{21}U}=\{(v_1^{(C_2)},u_2),(v_1^{(C_2)},u_3)\}$,

$X_{C_{23}U}=\{(v_3^{(C_2)},u_1),(v_3^{(C_2)},u_2)\}$,

$X_{C_3U}=X_{C_{32}U}=\{(v_2^{(C_3)},u_1),(v_2^{(C_3)},u_3)\}$,

$X_{PU}=\{(p^{(1)}_{1,2},u_3)\}$,

$L=L_1\cup L_2\cup L_3$,$L_1=\{(v_2^{(C_3)},u_2)\}$,

$L_2=\{(\bar{v}_1^{(V)},\bar{v}_3^{(V)})\}$,

$L_3=\{(p^{(1)}_{1,2},\bar{v}_1^{(C_1)}),(p^{(1)}_{1,2},v_2^{(C_1)}),(p^{(1)}_{1,2},v_3^{(V)})\}$.

X 可表示如图 4(以步骤为顶,互斥约束为边).

以用户为颜色对顶点着色,可表示各步骤的资源分配情况. 可行资源分配对应着相邻顶点颜色不同的着色方案. 按照定理 1 证明中所述对应和简化方法,可行资源分配数为 3,与 f 的成真赋值数相同. 设这 3 个可行资源分配为 π_1,π_2,π_3,三者均满足 $\pi_i(u_j)=u_j(1\leqslant i$

$\leqslant 3;0\leqslant j\leqslant 3$),如下给出它们其余的指派.

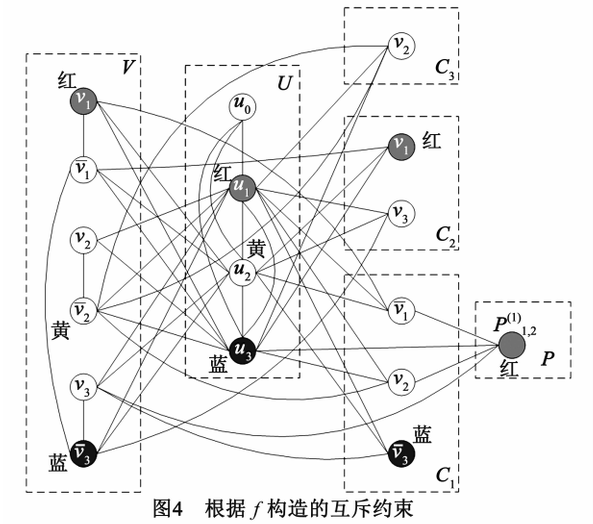


图4 根据 f 构造的互斥约束

$$①\pi_1(v_1^{(V)})=\pi_1(v_2^{(V)})=\pi_1(\bar{v}_3^{(V)})=\pi_1(v_1^{(C_2)})=\pi_1(v_2^{(C_3)})=\pi_1(v_2^{(C_1)})=\pi_1(\bar{v}_3^{(C_1)})=u_0,$$

$$\pi_1(\bar{v}_1^{(V)})=\pi_1(\bar{v}_1^{(C_1)})=u_1,$$

$$\pi_1(\bar{v}_2^{(V)})=\pi_1(p_{1,2}^{(1)})=u_2,$$

$$\pi_1(v_3^{(V)})=\pi_1(v_3^{(C_2)})=u_3.$$

$$②\pi_2(\bar{v}_1^{(V)})=\pi_2(v_2^{(V)})=\pi_2(v_3^{(V)})=\pi_2(\bar{v}_1^{(C_1)})=\pi_2(v_2^{(C_3)})=\pi_2(v_2^{(C_1)})=\pi_2(v_3^{(C_2)})=u_0,$$

$\pi_2(v_1^{(V)})=\pi_2(v_1^{(C_2)})=\pi_2(p_{1,2}^{(1)})=u_1$,($p_{1,2}^{(1)}$ 也可指派给 u_2,没有实质差异,如前述只取其一).

$$\pi_2(\bar{v}_2^{(V)})=u_2,$$

$$\pi_2(\bar{v}_3^{(V)})=\pi_2(\bar{v}_3{}^{(C_1)})=u_3.$$

π_2 的着色表示见图 4,步骤顶点着色为白、红、黄、蓝,表示其执行用户为 u_0、u_1、u_2、u_3.

$$③\pi_3(v_1^{(V)})=\pi_3(v_2^{(V)})=\pi_3(v_3^{(V)})=\pi_3(v_1^{(C_2)})=\pi_3(v_2^{(C_3)})=\pi_3(v_2^{(C_1)})=\pi_3(v_3^{(C_2)})=u_0,$$

$$\pi_3(\bar{v}_1^{(V)})=\pi_3(\bar{v}_1^{(C_1)})=u_1,$$

$$\pi_3(\bar{v}_2^{(V)})=\pi_3(p_{1,2}^{(1)})=u_2,$$

$$\pi_3(\bar{v}_3^{(V)})=\pi_3(\bar{v}_3^{(C_1)})=u_3.$$

4 结论与下一步工作

WS 是工作流访问控制的基本问题. 现有研究多局限于其决策问题,只能验证 AC 策略的正确性. 本文指出 WS 计数研究有利于提高 AC 规划的合理性,并对 #WS($\neq$,=),即互斥和绑定约束下的 WS 计数问题,进行研究,证明它属于#P 完全问题类,这说明:

(1)若#WS($\neq$,=)可在多项式时间内求解,则所有#P 问题可在多项式时间求解. 鉴于人们在现有#P 问题上付出了大量努力仍未做到这一点,#WS($\neq$,=)的多项式时间求解也是不太可能的.

(2)#WS($\neq$,=)与任何已知的#P 完全问题可在多项式时间内相互计数归约. 鉴于某些#P 完全问题已经存在优化的指数时间算法或实用求解器,有可能利用这些成果给出相对优化的#WS($\neq$,=)算法.

下一步将沿此思路研究#WS($\neq$,=)的优化求解.

参考文献

[1] AALST W, HEE K. Workflow Management: Models, Methods and Systems[M]. Massachusetts, USA: MIT Press, 2004.

[2] 翟治年,卢亚辉,奚建清,等. 面向企业级流程的职责分离框架及其冗余分析[J]. 电子学报,2013,41(10):2087-2093.
ZHAI Z, LU Y, XI J, et al. Enterprise-level business process oriented framework for separation of duties with its redundancy analysis[J]. Acta Electronica Sinica, 2013, 41(10): 2087-2093. (in Chinese)

[3] WANG Q, Li N. Satisfiability and resiliency in workflow authorization systems[J]. ACM Transactions on Information and System Security, 2010, 13(4): 1-35.

[4] BERTINO E, FERRARI E, ATLURI V. The specification and enforcement of authorization constraints in workflow management systems[J]. ACM Transactions on Information System Security, 1999, 2(1): 65-104.

[5] CRAMPTON J, GUTIN G, YEO A. On the parameterized complexity and kernelization of the workflow satisfiability problem[J]. ACM Transactions on Information and System Security, 2013, 16(1): 1-31.

[6] ATLURI V, BERTINO E, FERRARI E, et al. Supporting delegation in secure workflow management systems[A]. 17th Annual Conference on Data and Application of Security[C]. Berlin, GER: Springer, 2003. 190-202.

[7] CRAMPTON J, KHAMBHAMMETTU H. Delegation and satisfiability in workflow systems[A]. 13th ACM Symposium on Access Control Models and Technologies[C]. NY, USA: ACM, 2008. 31-40.

[8] BASIN D, BURRI S, KARJOTH G. Obstruction-free authorization enforcement: aligning security and business objectives[A]. Computer Security Foundations Symposium[C]. WA, USA: IEEE CS, 2011. 99-113.

[9] 赵相福,欧阳丹彤. 使用 SAT 求解器产生所有极小冲突部件集[J]. 电子学报,2009,37(4):804-810.
ZHAO X, OUYANG D. Deriving all minimal conflict sets using satisfiability algorithms[J]. Acta Electronica Sinica, 2009, 37(4): 804-810. (in Chinese)

[10] VALIANT L. The complexity of computing the permanent [J]. Theoretical Computer Science, 1979, 8(2): 189

-201.
[11] VALIANT L. The complexity of enumeration and reliability problems[J]. SIAM Journal on Computing, 1979, 8(3):410-421.
[12] DAHLLÖF V, JONSSON P, WAHLSTRÖM M. Counting models for 2sat and 3sat formulae[J]. Theoretical Computer Science, 2005, 332(1-3):265-291.
[13] BULATOV A. The complexity of the counting constraint satisfaction problem[J]. Journal of the ACM, 2013, 60(345):1-29.
[14] VADHAN S P. The Complexity of Counting[D]. MA, USA: Harvard University, 1995.

作者简介

翟治年　男,1977 年 5 月生,河北张家口人. 2000 年、2003 年和 2012 年分别在合肥工业大学土木建筑工程学院和华南理工大学计算机科学与工程学院取得工学学士、硕士和博士学位. 曾在广东北电通信设备有限公司从事 C ++ 开发. 现为浙江科技学院信息与电子工程学院讲师,主要从事访问控制、服务组合与工作流调度方面的研究.　E-mail:zhaizhinian@gmail.com

卢亚辉　男,1976 年 5 月生,陕西宝鸡人. 1997 年、2003 年和 2008 年分别在南京大学、中科院遥感所和清华大学获理学学士、理学硕士和工学博士学位. 现为深圳大学计算机与软件学院副教授,主要从事工作流、系统安全、Petri 网、Pi 演算等方面的研究.
E-mail:luyahui@szu.edu.cn

余法红(通讯作者)　男,1977 年 11 月生,湖北监利人. 2012 年毕业于武汉大学软件工程国家重点实验室计算机软件与理论专业,获工学博士学位. 现为嘉兴学院数理与信息工程学院讲师,主要从事复杂网络、数据挖掘与网络安全方面的研究.
E-mail:fhyu520@126.com