逻辑学研究　2019 年第 2 期，27–42
文章编号：1674-3202(2019)-02-0027-16

结构平衡理论的时态模型：形式系统与程序实现

王轶　　骆犀羚

摘　要：敌友逻辑（van der Hoek, *et al.*, 2018）采用结构平衡理论的视角对社会网络的动态变化进行了刻画。在一个稳定的社会网络中，主体之间没有理由改变当下的关系，而不稳定网络则通常会向稳定网络演进。敌友逻辑基于分支时间逻辑 CTL，其中每条时间线表示网络的一个演进过程。本文前半部分探讨敌友逻辑的可靠且完全的公理系统。敌友逻辑的模型检测、有效性和可满足性检测问题的计算复杂性已知都是 PSPACE 完全的。本文后半部分介绍敌友逻辑模型检测的程序实现。

关键词：敌友逻辑；结构平衡理论；时态逻辑；公理系统；模型检测

中图分类号：B81　　**文献标识码：**A

1　引言

对由网络结构导致的社会网络随时间演变的特性进行逻辑分析是*敌友逻辑*（Logic of Allies and Enemies）（[10]）的主要目的。其理论基础来自于社会心理学家提出的结构平衡理论（[8, 9]），而后又被推广为使用图论来进行刻画（[1, 5, 6]）。在该理论中，社会网络被处理为加标图：每个节点代表一个主体，每条边代表一个*连带*（tie）且标以*积极*或*消极*符号。积极符号通常用于标识朋友或盟友关系，而消极符号则用于敌人或对手关系。

如果网络结构满足特定的条件，那么这个网络是*平衡的*。举例来说，三个互为朋友的主体所表示的关系结构是平衡的，而三个互相敌对的主体所代表的关系结构被认为是不平衡的。经典文献（[1, 5–7]）中通过图形中的回路来定义网络的结构平衡性。如果一条回路中有偶数条消极边，则该回路是积极的；反之如果有奇数条边，则整条回路是消极的。例如，图 1 表示具有五个主体（a, b, c, d, e）和三

收稿日期：2018-11-13
作者信息：王轶　浙江大学哲学系
ynw@xixilogic.org
骆犀羚　浙江大学哲学系
xiling@zju.edu.cn
基金项目：国家社会科学基金项目（16CZX048、18ZDA290）。

条回路（abc、$abdec$ 和 $bced$）的网络。因为图中每条回路都有偶数个消极边，所以全部回路都是积极的。这种只有积极回路的网络就称为是平衡的。结构平衡理论认为，社会网络有向平衡网络演变的趋势，虽然这一过程可能需要很长时间，并且可能由于各方面因素的影响而无法真正达到平衡。

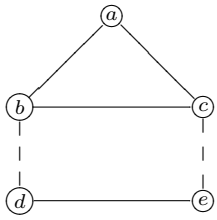


图 1: 平衡网络示意图，其中，实线和虚线分别表示积极和消极连带

敌友逻辑引入了一个与结构平衡秉承相似直观但有所不同的概念，称为*稳定性*。在敌友逻辑中，如果一对主体更愿意保持彼此现有的关系，那么他们当下的关系是稳定的。如果网络中每对主体间的关系都是稳定的，那么整个网络就是稳定的。更细致地说，如果两个主体是朋友，并且他们的*共识*（即共同的朋友或敌人）多于*分歧*（即与其一为友而与另一为敌的那些主体），则两者的关系是稳定的。如果两个主体是敌人，并且分歧多于共识，该关系也是稳定的。此外，对于非敌非友的两个主体，如果共识与分歧数量相等，则也具有稳定关系。其他情况则是不稳定的。

稳定性概念乍看起来或许与平衡性相去甚远，但它采用的观点实际上源自结构平衡理论最初的想法 （[8, 9]）：对第三者抱有相似态度的一对个体倾向于彼此喜欢，对第三者抱有不同态度的一对个体倾向于彼此厌恶。

文中将引入敌友逻辑的公理系统，它在分支时间逻辑 CTL（[3]）的公理系统之上增加一些与网络相关的公理和规则。该系统的可靠性和完全性也将得到证明。

敌友逻辑的模型检测、有效性以及可满足性检测问题都是 PSPACE 完全的（[10]），其具体算法已通过 Python 程序实现，后文将具体说明其主要功能，包括如下常用函数：

- `staScore()`：对于给定的网络，计算其稳定性评分
- `isStable()`：判断给定网络是否稳定
- `succNum()`：对于给定的网络，计算其后继数目
- `successors()`：返回存储给定网络所有后继的列表
- `isSucc()`：判断一网络是否为另一网络的后继
- `mc()`：对于给定的网络和给定的公式，判断该网络是否满足该公式
- `sat()` 和 `valid()`：对于给定的公式，分别判断该公式是否可满足或是否

有效

- `satFind()`：基于给定情况，找到满足给定公式的网络

文章结构如下。第 2 节介绍敌友逻辑 （[10]）的语法和语义。第 3 节引入敌友逻辑的公理化，并证明其可靠性和完全性。第 4 节介绍敌友逻辑模型检测、有效性检测和可满足性检测的程序实现；其中第 4.1 和 4.2 节分别涵盖语义（网络相关）和语法（句法检查）的程序实现，而对模型、有效性和可满足性检测程序的说明则在第 4.3 节。文章最后一节总结并讨论后续工作。程序代码、例子和后期更新将通过如下网站维护：`http://www.xixilogic.org/projects/lae`。

2　敌友逻辑

本节简要介绍敌友逻辑 （[10]）的语法、语义及其模型检测等问题。以自然数表示主体，ω 表示所有主体的集合。当自然数 n 用于表示网络的个体与或者语言的参数时，它代表一个自然数的集合（即集合 $\{0, 1, \ldots, n-1\}$）。这种处理方式与公理集合论中对自然数的常见定义保持一致（不妨参考 [11]）。

2.1　网络与稳定性

n 节点网络是具有 n 个节点的带三个符号的有穷无向完全图。准确地讲，一个 n 节点网络是一个有序对 (n, E)，使得：

- $n = \{0, \ldots, n-1\}$ 是有穷的主体集；
- $E : \{\{i, j\} \mid i, j \in n\} \to \{1, 0, -1\}$ 是一个全函数，用于表示每对主体之间的积极 (+)、消极 (−) 或中立 (0) 连带，并且满足 $E(\{i, i\}) = 0$。

这里不考虑自反圈（例如 i 是自己的朋友或敌人）或两个主体之间具有多重关系（例如 i 和 j 既是朋友又是敌人）等情形。在不引起混淆的情况下，通常以 ij 或 ji 表示边 $\{i, j\}$。有时也以 $E(ij)$ 或 $E(ji)$（或者 $E(i, j)$ 或 $E(j, i)$）表示 $E(\{i, j\})$。

在敌友逻辑中，两个主体之间的*共识*或*吸引力*定义为使他们建立或保持朋友关系的理由数，即他们的共同朋友或敌人的总数。类似的，两个主体间的*分歧*或*排斥力*是使他们建立或保持敌对关系理由数，即与其一为友但与另一主体为敌的主体的总数。准确说来，令 $N = (n, E)$ 为一网络且 $i, j \in n$，则 ij 之间的共识（记为 $attr(ij)$）和分歧（记为 $rep(ij)$）满足：

$$
\begin{aligned}
attr(ij) &= |\{k \mid E(ik) = E(jk) = 1\}| + |\{k \mid E(ik) = E(jk) = -1\}| \\
rep(ij) &= |\{k \mid E(ik) = 1 \text{ 且} E(jk) = -1\}| + |\{k \mid E(ik) = -1 \text{ 且} E(jk) = 1\}|
\end{aligned}
$$

两主体间关系的*稳定性*取决于他们现有的关系和彼此间的共识与分歧。如果

i 和 j 是朋友（即 $E(ij)=1$），且他们之间的共识不小于分歧，则他们的关系是稳定的。类似地，如果 i 和 j 是敌人，且彼此间分歧不小于共识，则也是稳定的。而中立关系只有当吸引力等于排斥力时才是稳定的。除此以外的情况都是不稳定的。ij 连带的稳定性评分以 $score(ij)$ 表示，定义如下：

$$score(ij)=\begin{cases} attr(ij)-rep(ij), & \text{如果 } E(ij)=1 \\ rep(ij)-attr(ij), & \text{如果 } E(ij)=-1 \\ -|attr(ij)-rep(ij)|, & \text{如果 } E(ij)=0 \end{cases}$$

不难发现，如果 $score(ij)\geq 0$，那么 ij 连带是稳定的。反之，则不稳定。

如果网络中所有的边都稳定，则称该网络稳定。反之则不稳定。一个网络的稳定性评分是其中所有边的稳定性评分之和。

称网络 $N_2=(n,E_1)$ 是网络 $N_1=(n,E_2)$ 的后继（记为 $N_1\rightsquigarrow N_2$），如果满足如下条件之一：

- 当 N_1 稳定时：$N_1=N_2$,
- 当 N_1 不稳定时：N_1 与 N_2 仅在一条边 ij 的符号上有差别，同时满足：要么在 N_1 中有 $attr(ij)>rep(ij)$ 且 $E_2(ij)=1$，要么在 N_1 中有 $rep(ij)>attr(ij)$ 且 $E_2(ij)=-1$。

网络集上的后继关系形成了一个树形结构，其每条分支代表网络随时间演进的过程，仅有的循环出现于稳定网络到自身的回路。关于上述定义和结论的更多解释详见（[10]）。这里值得强调的是，网络的演进过程中稳定性评分并不一定递增。虽然在多数情况下，后继网络的稳定性评分确实高于其前驱，并且这确实反映了网络结构趋于平衡的性质，但并不能排除稳定性评分走低的临时情况。

2.2 语言和语义

这里引入有穷版本的 LAE，即可用主体集为某个自然数 $n=\{0,1,\ldots,n-1\}$，记为 LAE^n。参数 n 可以是任意大于等于 3 的自然数（当 $n<3$ 时网络平衡概念失去意义，因此不予考虑）。LAE^n 的语言 $\mathcal{L}^n$ 的语法定义如下：

$$\begin{aligned}\varphi ::= {} & \mathrm{P}(i,i)\mid \mathrm{N}(i,i)\mid \mathrm{O}(i,i)\mid \mathsf{stb}\mid \mathsf{bal}\mid \mathrm{T}\mid \sim\varphi\mid /\backslash(\varphi,\varphi)\mid \backslash/(\varphi,\varphi)\mid \text{->}(\varphi,\varphi)\\ & \text{<->}(\varphi,\varphi)\mid \mathrm{AX}\varphi\mid \mathrm{EX}\varphi\mid \mathrm{AG}\varphi\mid \mathrm{EG}\varphi\mid \mathrm{AF}\varphi\mid \mathrm{EF}\varphi\mid \mathrm{AU}(\varphi,\varphi)\mid \mathrm{EU}(\varphi,\varphi)\end{aligned}$$

其中 $i\in n$。原子命题 $\mathrm{P}(i,j)$、$\mathrm{N}(i,j)$ 和 $\mathrm{O}(i,j)$ 将分别用于表示 ij 边是积极、消极和中立的，而 stb 和 bal 将分别表示网络是稳定和平衡的。符号 T、$\sim$、/\、\/、-> 和 <–> 是命题联结词（正文中使用中缀形式 $\wedge$、$\vee$、$\rightarrow$ 和 $\leftrightarrow$ 分别表示后四

个符号以方便阅读)。其他算子来自 CTL （[2, 3])。

$\mathcal{L}^n$ 通过大小为 n 的网络的集合加以解释。任意 $N=(n,E)$ 是否满足（记作 $\models$）某个 $\mathcal{L}^n$ 公式可归纳定义如下（其中 $\varphi,\psi\in\mathcal{L}^n$ 且 $i,j\in n$)：

$$
\begin{array}{ll}
N\models \mathrm{P}(i,j) & \Longleftrightarrow\ E(ij)=1\\
N\models \mathrm{N}(i,j) & \Longleftrightarrow\ E(ij)=-1\\
N\models \mathrm{O}(i,j) & \Longleftrightarrow\ E(ij)=0\\
N\models \mathsf{stb} & \Longleftrightarrow\ N\text{ 是稳定的}\\
N\models \mathsf{bal} & \Longleftrightarrow\ N\text{ 是平衡的}\\
N\models \mathrm{T} & \text{恒成立}\\
N\models \sim\varphi & \Longleftrightarrow\ \text{并非 } N\models\varphi\\
N\models /\backslash(\varphi,\psi) & \Longleftrightarrow\ N\models\varphi\text{ 且 }N\models\psi\\
N\models \backslash/(\varphi,\psi) & \Longleftrightarrow\ N\models\varphi\text{ 或 }N\models\psi\\
N\models ->(\varphi,\psi) & \Longleftrightarrow\ N\models\varphi\text{ 实质蕴涵 }N\models\psi\\
N\models <->(\varphi,\psi) & \Longleftrightarrow\ N\models ->(\varphi,\psi)\text{ 且 }N\models ->(\psi,\varphi)\\
N\models \mathrm{AX}\varphi & \Longleftrightarrow\ N\text{ 的任一后继 }N'\text{（即满足 }N\rightsquigarrow N'\text{），都有 }N'\models\varphi\\
N\models \mathrm{EX}\varphi & \Longleftrightarrow\ N\text{ 的某一后继 }N'\text{ 满足 }N'\models\varphi\\
N\models \mathrm{AG}\varphi & \Longleftrightarrow\ \text{任给时间线 }N=N_0\rightsquigarrow N_1\rightsquigarrow\cdots\text{，任给 }i\text{ 都有 }N_i\models\varphi\\
N\models \mathrm{EG}\varphi & \Longleftrightarrow\ \text{存在时间线 }N=N_0\rightsquigarrow N_1\rightsquigarrow\cdots\text{，任给 }i\text{ 都有 }N_i\models\varphi\\
N\models \mathrm{AF}\varphi & \Longleftrightarrow\ \text{任给时间线 }N=N_0\rightsquigarrow N_1\rightsquigarrow\cdots\text{，存在 }i\text{ 使得 }N_i\models\varphi\\
N\models \mathrm{EF}\varphi & \Longleftrightarrow\ \text{存在时间线 }N=N_0\rightsquigarrow N_1\rightsquigarrow\cdots\text{，存在 }i\text{ 使得 }N_i\models\varphi\\
N\models \mathrm{AU}(\varphi,\psi) & \Longleftrightarrow\ \text{任给时间线 }N=N_0\rightsquigarrow N_1\rightsquigarrow\cdots\text{，存在 }i\text{ 使得}\\
 & \qquad N_i\models\psi\text{ 且对任意 }j<i\text{，}N_j\models\varphi\\
N\models \mathrm{EU}(\varphi,\psi) & \Longleftrightarrow\ \text{存在时间线 }N=N_0\rightsquigarrow N_1\rightsquigarrow\cdots\text{，存在 }i\text{ 使得}\\
 & \qquad N_i\models\psi\text{ 且对任意 }j<i,N_j\models\varphi
\end{array}
$$

语句“ij 边稳定”（记为 $\mathsf{stb}(i,j)$）可以在 $\mathcal{L}^n$ 中表达如下：

$$(\mathrm{P}(i,j)\wedge\mathrm{AXP}(i,j))\vee(\mathrm{N}(i,j)\wedge\mathrm{AXN}(i,j))\vee(\mathrm{O}(i,j)\wedge\mathrm{AXO}(i,j))$$

因此不必作为原子语句引入。

由上述语义可以得到一系列逻辑 LAE^n ($n\in\mathbb{N}$ 且 $n\geq 3$)，通常以 LAE 表示任意这样的逻辑。如下定义引自 （[10])：

定理 1 (LAE 的计算复杂性)**.** *LAE 的模型检测与有效性检测问题都是 PSPACE 完全的。*

3 公理系统

由定理 1，敌友逻辑 LAE 的有效性检测问题是可判定的，因此是可公理化的：至少可以穷举所有有效式。但这并不意味着可以得到一个契合语法证明直观的公理系统。本节给出 LAE 的公理系统，并证明其可靠性和完全性。

下文将给出逻辑 LAE^n 的公理系统 **LAE**^{n}，其中引入了*描述*这一概念。*描述*（将以 δ 表示）是形如 $\bigwedge_{i,j\in n} p_{i,j}$ 的公式，其中 $p_{i,j}$ 是原子命题 $\mathrm{P}(i,j)$、$\mathrm{N}(i,j)$ 或 $\mathrm{O}(i,j)$。描述将用于刻画网络，而最好的方式是使同一网络被唯一的描述所刻画。为实现这一目的，需要增加一些限制条件。

首先约定描述中的边按照特定顺序排列，例如规定 $p_{1,2}$ 永远出现在 $p_{1,3}$ 的左边，这样可以避免出现对同一网络的逻辑等值的不同描述。令 $N=(n,E)$ 为一网络，N 的*描述*，记为 $\delta(N)$，是满足如下要求的合取式 $\bigwedge_{i,j\in n} p_{i,j}$（合取项有确定的顺序）：

$$p_{i,j}=\begin{cases}\mathrm{P}(i,j), & \text{如果 } E(ij)=1\\ \mathrm{N}(i,j), & \text{如果 } E(ij)=-1\\ \mathrm{O}(i,j), & \text{如果 } E(ij)=0\end{cases}$$

在上述限制条件下，不难证明 $\delta(N)$ 是存在且唯一的。此外，并非所有的描述都是某个网络的描述（例如出现合取支 $\mathrm{P}(1,1)$，或同时出现 $\mathrm{P}(1,2)$ 和 $\mathrm{N}(2,1)$ 等情况），但如果 δ 是 N 的描述，即 $\delta=\delta(N)$，则称 δ 描述了 N。不难证明，$\delta(N)$ 描述了唯一一个网络，即 N。

稳定公理（记为 SF）是如下公式：

$$(\mathrm{SF}) \qquad \mathsf{stb}\leftrightarrow(\delta_1\vee\cdots\vee\delta_m)$$

其中 $\delta_1,\ldots,\delta_m$ 枚举了所有稳定网络的描述。类似地，*平衡公理*（记为 BF）是如下公式：

$$(\mathrm{BF}) \qquad \mathsf{bal}\leftrightarrow(\delta_1\vee\cdots\vee\delta_m)$$

其中，$\delta_1,\ldots,\delta_m$ 枚举了所有平衡网络的描述。

令 δ 描述网络 N，则 δ 的*明日公理*（记为 $tf(\delta)$）是如下公式:

$$tf(\delta) \qquad \delta\rightarrow(\mathrm{AX}(\delta_1\vee\cdots\vee\delta_m)\wedge\mathrm{EX}\delta_1\wedge\cdots\wedge\mathrm{EX}\delta_m)$$

其中，$\delta_1,\ldots,\delta_m$ 是 δ 所描述的网络的所有后继的描述。明日公理 $tf(\delta)$ 可以简写为 $\delta\rightarrow\nabla\{\delta_1,\ldots,\delta_m\}$，其中的 ∇ 有时称为*覆盖模态词*（cover modality）。明日公理的数量是有穷的。

逻辑 LAE^n 的公理系统 **LAE**^{n} 如图 2 所示。**LAE**^{n} 由 CTL 的公理和规则（[4]）、用于刻画网络以及网络更新的公理和规则以及一些 CTL 时态算子的定义公理组成。{AX, AU, EU} 是 CTL 时态算子的一个完全集，因此其他算子 EX、AG、EG、AF 和 EF 都是可定义的，这些定义公理都放在“CTL 模态词定义”栏目下。此外，CTL 公理 D_X 可由明日公理推导出，因此可以删去。

CTL 公理和规则：

(PC)	所有命题重言式的代入
(K_X)	$AX(\varphi \to \psi) \to (AX\varphi \to AX\psi)$
(D_X)	EXT
(EU)	$EU(\varphi, \psi) \leftrightarrow (\psi \vee (\varphi \wedge EXEU(\varphi, \psi)))$
(AU)	$AU(\varphi, \psi) \leftrightarrow (\psi \vee (\varphi \wedge AXAU(\varphi, \psi)))$
(MP)	由 φ 和 $\varphi \to \psi$ 得到 ψ
(Nec)	由 φ 得到 $AX\varphi$
(E-ind)	由 $(\psi \vee (\varphi \wedge EX\chi)) \to \chi$ 得到 $EU(\varphi, \psi) \to \chi$
(A-ind)	由 $(\psi \vee (\varphi \wedge AX\chi)) \to \chi$ 得到 $AU(\varphi, \psi) \to \chi$

CTL 模态词定义：

(EX)	$EX\varphi \leftrightarrow \sim AX \sim \varphi$
(AF)	$AF\varphi \leftrightarrow AU(T, \varphi)$
(EF)	$EF\varphi \leftrightarrow EU(T, \varphi)$
(AG)	$AG\varphi \leftrightarrow \sim EF \sim \varphi$
(EG)	$EG\varphi \leftrightarrow \sim AF \sim \varphi$

与网络相关的公理和规则：

(SF)	稳定公理
(BF)	平衡公理
(TF)	所有明日公理
(Symm)	$(P(i,j) \leftrightarrow P(j,i)) \wedge (N(i,j) \leftrightarrow N(j,i))$
(PN)	$P(i,j) \to \sim N(i,j)$
(Neut)	$O(i,j) \leftrightarrow (\sim P(i,j) \wedge \sim N(i,j))$
(Tm1)	由 $\delta \to \mathsf{stb}$ 得到 $\delta \to (AU(\varphi, \psi) \to \psi)$
(Tm2)	由 $\delta \to \mathsf{stb}$ 得到 $\delta \to (EU(\varphi, \psi) \to \psi)$

图 2: 公理系统 **LAE**^{n}，其中 φ、ψ 和 χ 代表任一公式，δ 代表任一网络的描述，且 i 和 j 是小于 n 的自然数。

不难验证 **LAE**^{n} 的可靠性：所有公理都有效并且所有规则都保持有效性。其具体证明不再赘述。这里主要关注 **LAE**^{n} 的完全性。下面的结论适用于任意数量的主体（即任意 $n \in \omega$ 且 $n \geq 3$），因此以 $\Phi \vdash \varphi$ 表示公式 φ 是以公式集 Φ 为前提在系统 **LAE**^{n} 中可推演的（$\vdash \varphi$ 即表示 φ 是 **LAE**^{n} 的定理）。

引理 1. 假设 $\vdash \delta \to \nabla\{\delta_1, \ldots, \delta_m\}$，则如下结论成立：

1. 如果对所有 $i = 1, \ldots, m$ 都有 $\vdash \delta_i \to \varphi$，则 $\vdash \delta \to AX\varphi$

2. 如果存在 $i = 1, \ldots, m$ 使得 $\vdash \delta_i \to {\sim}\varphi$，则 $\vdash \delta \to {\sim}AX\varphi$

证明. 1. 不难发现公式 $((\psi \to \nabla\{\psi_1, \ldots, \psi_m\}) \wedge \bigwedge_{i=1}^{m} \mathrm{AX}(\psi_i \to \varphi)) \to (\psi \to \mathrm{AX}\varphi)$ 是 CTL 的有效式。根据 CTL 的完全性，它可由 CTL 的公理与规则推出。所以，该公式是 $\mathbf{LAE}^n$ 的定理。对任意 $i = 1, \ldots, m$，由前提 $\vdash \delta_i \to \varphi$ 和公理 (Nec) 可以得到 $\vdash \mathrm{AX}(\delta_i \to \varphi)$。再由假设 $\vdash \delta \to \nabla\{\delta_1, \ldots, \delta_m\}$ 和上述 $\mathbf{LAE}^n$ 定理，就可以得到 $\vdash \delta \to \mathrm{AX}\varphi$。引理的第 2 项可以使用类似的方法，通过 CTL 有效式 $((\psi \to \nabla\{\psi_1, \ldots, \psi_m\}) \wedge \bigvee_{i=1}^{m} \mathrm{AX}(\psi_i \to {\sim}\varphi)) \to (\psi \to {\sim}\mathrm{AX}\varphi)$ 加以证明。 □

引理 2. 对任意网络的描述 δ 和任意公式 φ，要么 $\vdash \delta \to \varphi$，要么 $\vdash \delta \to {\sim}\varphi$。

证明. 网络的描述中的两个不同合取支不能同时为集合 $\{\mathrm{P}(i,j), \mathrm{N}(i,j), \mathrm{O}(i,j)\}$ 中的元素。这由公理 (PN) 和 (Neut) 保证。此外，$\mathrm{P}(i,j)$ 和 $\mathrm{P}(j,i)$（对于 N 和 O 也类似）要么同为网络描述的两个合取支，要么都不是，这由公理 (Symm) 和 (Neut) 保证。因此，网络描述的一致性可以通过命题逻辑和这些公理得到。因此 $\vdash \delta \to \varphi$ 和 $\vdash \delta \to {\sim}\varphi$ 不能同时成立。

下面通过对 φ 做结构归纳来证明 $\vdash \delta \to \varphi$ 与 $\vdash \delta \to {\sim}\varphi$ 中至少有一个成立。

- φ 是原子公式 $\mathrm{P}(i,j)$。如果 $\mathrm{P}(i,j)$ 是 δ 的一个合取支，那么 $\vdash \delta \to \mathrm{P}(i,j)$。否则 $\mathrm{N}(i,j)$ 和 $\mathrm{O}(i,j)$ 之一是 δ 的合取支。因此，要么 $\vdash \delta \to \mathrm{N}(i,j)$，要么 $\vdash \delta \to \mathrm{O}(i,j)$。又因为 $\vdash \mathrm{N}(i,j) \to {\sim}\mathrm{P}(i,j)$（根据 PN）且 $\vdash \mathrm{O}(i,j) \to {\sim}\mathrm{P}(i,j)$（根据 Neut），故有 $\vdash \delta \to {\sim}\mathrm{P}(i,j)$。
- φ 为 $\mathrm{N}(i,j)$ 和 $\mathrm{O}(i,j)$ 的情形可作类似证明。
- $\varphi = {\sim}\psi$：若 $\nvdash \delta \to {\sim}\psi$，由归纳假设知 $\vdash \delta \to \psi$，故 $\vdash \delta \to {\sim}{\sim}\psi$。
- $\varphi = (\psi \to \chi)$：若 $\nvdash \delta \to (\psi \to \chi)$，则 $\nvdash \delta \to {\sim}\psi$ 且 $\nvdash \delta \to \chi$。因此，$\nvdash \delta \to ({\sim}\psi \vee \chi)$，即 $\nvdash \delta \to (\psi \to \chi)$。
- 其他命题联结词情形可类似证明。
- φ 为 stb 的情形。假设 $\nvdash \delta \to \mathsf{stb}$。在此情况下 δ 不可能描述稳定网络。否则根据稳定性公理，$\vdash \mathsf{stb} \leftrightarrow (\delta_1 \vee \cdots \vee \delta_m)$（其中 $\delta_1, \ldots, \delta_m$ 是所有稳定网络的描述）。如果 δ 描述稳定网络，那么 δ 是某个 δ_i（$i = 1, \ldots, m$）。于是 $\vdash \delta \to (\delta_1 \vee \cdots \vee \delta_m)$，并因此 $\vdash \delta \to \mathsf{stb}$，与假设矛盾。所以 δ 描述的不是稳定网络，并且 δ 与所有的 δ_i 都至少有一个合取支是不同的。由此可知，$\vdash \delta \to {\sim}(\delta_1 \vee \cdots \vee \delta_m)$。否则存在 $i = 1, \ldots, m$ 使得 $\vdash \delta \to \delta_i$，矛盾。根据稳定性公理，$\vdash \delta \to {\sim}\mathsf{stb}$。
- φ 为 $\mathrm{AX}\psi$ 的情形。假设 $\nvdash \delta \to \mathrm{AX}\psi$。由明日公理，有 $\vdash \delta \to \nabla\{\delta_1, \ldots, \delta_m\}$（其中 $\delta_1, \ldots, \delta_m$ 是 δ 所描述网络的所有后继的描述）。在根据引理 1(1)，存在 $i = 1, \ldots, m$ 使得 $\nvdash \delta_i \to \psi$。由归纳假设知 $\vdash \delta_i \to {\sim}\psi$。故由引理 1(2)，$\vdash \delta \to {\sim}\mathrm{AX}\psi$。

• φ 为 AU(ψ,χ) 的情形。假设 $\nvdash \delta \to \text{AU}(\psi,\chi)$。通过 (AU) 进行等值替换，有 $\nvdash \delta \to (\chi \vee (\psi \wedge \text{AXAU}(\psi,\chi)))$。由此可知，(1) $\nvdash \delta \to \chi$ 且 [(2) $\nvdash \delta \to \psi$ 或 (3) $\nvdash \delta \to \text{AXAU}(\psi,\chi)$]。当 (1) 和 (2) 同时成立时，根据归纳假设，有 $\vdash \delta \to {\sim}\chi$ 和 $\vdash \delta \to {\sim}\psi$。由此可知 $\vdash \delta \to {\sim}(\chi \vee (\psi \wedge \text{AXAU}(\psi,\chi)))$。再由 (AU)，有 $\vdash \delta \to {\sim}\text{AU}(\psi,\chi)$。当 (1) 和 (3) 同时成立时，由 (3) 和引理 1(1) 知，明日公理 $\delta \to \nabla\{\delta_1,\ldots,\delta_m\}$ 中存在 δ_i $(1 \le i \le m)$ 使得 $\nvdash \delta_i \to \text{AU}(\psi,\chi)$。这就得到与开头假设部分相似的情形，但是建立在 δ 的后继 δ_i 之上。继续这个过程有穷多次（明日公理有穷）直到得到 $\nvdash \delta^s \to \text{AU}(\psi,\chi)$，其中 δ^s 描述稳定网络（即 $\vdash \delta^s \to \mathsf{stb}$）。由此可得 $\nvdash \delta^s \to (\chi \vee (\psi \wedge \text{AXAU}(\psi,\chi)))$，同理亦可得 (1′) $\nvdash \delta^s \to \chi$ 且 [(2′) $\nvdash \delta^s \to \psi$ 或 (3′) $\nvdash \delta^s \to \text{AXAU}(\psi,\chi)$]。对于 (1′) 和 (2′) 同时成立的情形，仍有 $\vdash \delta^s \to {\sim}\chi$ 且 $\vdash \delta^s \to {\sim}\psi$，并且 $\vdash \delta^s \to {\sim}\text{AU}(\psi,\chi)$ 成立。通过反复使用 (AU) 进行等价替换，可得 $\vdash \delta \to {\sim}\text{AU}(\psi,\chi)$。当 (1′) 和 (3′) 同时成立时，因为 $\delta^s \to \nabla\delta^s$ 是明日公理，根据 (1') 和 (Tm1)，有 $\vdash \delta^s \to {\sim}\text{AU}(\psi,\chi)$。反复使用引理 1(2) 并使用 (AU) 进行等价替换，可以得到 $\vdash \delta \to {\sim}\text{AU}(\psi,\chi)$。

• $\varphi = \text{EU}(\psi,\chi)$ 的情况与 AU(ψ,χ) 类似，使用公理 (Tm2) 替代 (Tm1) 即可。

根据“CTL 模态词定义”类型中的公理，包含其他时态算子的公式可归约为不含这些算子的公式，从而结论对所有公式成立。 □

引理 3. *任给极大一致的公式集 Φ，描述 δ 和公式 φ。如果 $\delta,\varphi \in \Phi$，则 $\vdash \delta \to \varphi$。*

证明. 使用反证法，假设 $\delta,\varphi \in \Phi$ 且 $\nvdash \delta \to \varphi$。由引理 2 知 $\vdash \delta \to {\sim}\varphi$。根据 Φ 的极大性，有 ${\sim}\varphi \in \Phi$，从而 Φ 不一致。矛盾！ □

任给极大一致的 $\mathcal{L}^n$ 公式集 Φ，以 N^Φ 表示网络 (n,E)，其中 E 满足：

$$E(ij) = \begin{cases} 1, & \text{如果 } \text{P}(i,j) \in \Phi \\ -1, & \text{如果 } \text{N}(i,j) \in \Phi \\ 0, & \text{否则} \end{cases}$$

可以证明 N^Φ 存在且唯一。

引理 4. *令 Φ 为极大一致的公式集，且 δ 是网络的描述。则如下命题等价：*
1. $N^\Phi \models \delta$　　2. δ 描述 N^Φ　　3. $\delta \in \Phi$

证明. 1 和 2 等价由定义可得。这里证明 2 和 3 等价。通过定义不难由 2 得到 3；考虑 3 至 2 方向。假设 $\delta \in \Phi$。因为 δ 是 n 节点网络的描述，形如 $\bigwedge_{i,j=0}^{n-1} X(i,j)$（其中 $X \in \{\text{P},\text{N},\text{O}\}$）。对任意主体 i 和 j 而言，存在 $X(i,j)$ 是 δ 的合取支并属于 Φ，且不存在 $Y(i,j) \in \Phi$ 使得 $X \ne Y$。根据定义，δ 的这些合取支决定 N^Φ 所有边的标签，即 δ 描述 N^Φ。 □

引理 5. 令 Φ 为极大一致的公式集，则 $N^{\Phi} \models \Phi$。

证明. 假设 δ 描述 N^{Φ}。由引理 4 可得 $\mathsf{N}^{\Phi} \models \delta$ 和 $\delta \in \Phi$。对任意 $\varphi \in \Phi$，由引理 3，有 $\vdash \delta \to \varphi$。再由 $\mathbf{LAE}^n$ 的可靠性知 $\models \delta \to \varphi$，故 $\mathsf{N}^{\Phi} \models \varphi$。因此 $\mathsf{N}^{\Phi} \models \Phi$。 □

与 CTL 只有弱完全性不同，$\mathbf{LAE}^n$ 具有强完全性。这主要是因为在 $\mathbf{LAE}^n$ 中所有的时间线都是有穷的（每个网络都能在有限步到达稳定，且稳定网络不再发生变化）。

定理 2 (强完全性)**.** 任给极大一致的公式集 Φ 和公式 φ，如果 $\Phi \models \varphi$，则 $\Phi \vdash \varphi$。

4 程序实现

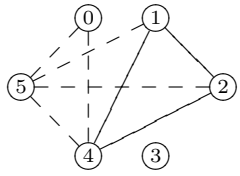
本节介绍基于 Python 的 LAE 模型和有效性检测程序，主要分为网络的实现（语义角度）、语法检测（语形角度）和模型检测等的实现（语义和语形交叉）三个部分。

4.1 网络的实现

在程序实现中，n 节点网络（$n \in \omega$）是二元组 (n, E)，其中 $E : n \times n \to \{-1, 0, 1\}$ 满足 $E(i, j) = E(j, i)$，且对所有 $i, j \in n$ 都有 $E(i, i) = 0$。该定义与第 2 节中的差别在于这里的每个主体对都是有序的，但因为保证了对称性，两个定义是共通的。

n 节点网络可以看成是 $n \times n$ 矩阵，其中每个元素 (i, j) 代表主体 i 与 j 之间的连带。左下图矩阵表示一个 6 节点网络，对应于右下图所示网络。

$$\begin{bmatrix} 0 & -1 & 0 & 0 & -1 & -1 \\ -1 & 0 & 1 & 0 & 1 & -1 \\ 0 & 1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 1 & 0 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 0 \end{bmatrix}$$



如前所述，网络示意图中的实线表示积极连带，虚线表示消极连带，而中性连带则不画线。该表示法与经典文献 （[1]）中一致。

4.1.1 生成和表示

对于程序实现而言，网络其实就是前面定义的矩阵。但程序同时支持通过指定网络中的积极和消极连带来生成网络。

netInit() 与 netGen() 这两个函数用于生成网络。函数 netInit(n) 初始化一个所有的边都是中性连带的 n 节点网络，即大小为 $n \times n$、元素全为 0 的矩阵。函数 netGen(net, sign, *edges) 可以通过为列表 *edges 中的边规定符号 sign 来修改网络 net。

例如，以下几行代码生成上面图形中给出的 6 节点网络，该网络命名为 net1（可使用命令 print(net1) 进行查看）。

```
net1 = netInit(6)
net1 = netGen(net1,1,(1,2),(1,4),(2,4))
net1 = netGen(net1,-1,(0,1),(0,4),(0,5),(1,5),(2,5),(4,5))
```

函数 netSize() 返回所输入网络的大小。如果对上面定义的网络 net1 执行命令 netSize(net1)，则返回值为 6。

4.1.2 评分计算

对于给定的网络，边 (i,j) 的共识、分歧和稳定性评分分别由函数 attr()、rep() 和 score() 给出：

- attr(net,i,j)：返回网络 net 中边 (i,j) 的共识；
- rep(net,i,j)：返回网络 net 中边 (i,j) 的分歧；
- score(net,i,j)：返回网络 net 中边 (i,j) 的稳定性评分。

如果省略参数 i 与 j，则输出结果为一个矩阵，其右上部每个位置的值都为对应边的相应值。例如，attr(4net)（其中，4net 是一个 4 节点网络）的结果如下：

$$\begin{bmatrix} 0 & \text{attr(4net,0,1)} & \text{attr(4net,0,2)} & \text{attr(4net,0,3)} \\ 0 & 0 & \text{attr(4net,1,2)} & \text{attr(4net,1,3)} \\ 0 & 0 & 0 & \text{attr(4net,2,3)} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

需要注意的是，对于在自循环 ii，共识为包含 i 的全部连带的数量，而分歧为 0。这些特殊情况目前在程序中并考虑，但不会导致后续检测结论出现错误。

用第 4.1.1 节中的例子，执行 print(attr(net1))、print(rep(net1)) 和 print(score(net1)) 命令将分别得到如下矩阵：

```
[[0 1 1 0 1 2]  [[0 1 2 0 1 0]  [[ 0  0 -1  0  0 -2]
 [0 0 2 0 3 1]   [0 0 0 0 0 2]   [ 0  0  2  0  3  1]
 [0 0 0 0 2 0]   [0 0 0 0 0 2]   [ 0  0  0  0  2  2]
 [0 0 0 0 0 0]   [0 0 0 0 0 0]   [ 0  0  0  0  0  0]
 [0 0 0 0 0 1]   [0 0 0 0 0 2]   [ 0  0  0  0  0  1]
 [0 0 0 0 0 0]]  [0 0 0 0 0 0]]  [ 0  0  0  0  0  0]]
```

网络的稳定性评分被定义为其所有边稳定性评分的总和。函数 staScore()（勿与 score() 混淆）可用于获得网络的稳定性评分。例如，命令 staScore(net1) 返回 4。

4.1.3 后继

函数 isStable() 用于检测一个网络是否稳定。函数 succNum() 返回给定网络所有不同后继的数量。采用第 4.1.1 和 4.1.2 节中的例子，succNum(net1) 返回 2，即网络 net1 有两个后继。

函数 successors() 返回给定网络的所有后继网络的列表。例如，执行命令 successors(net1) 将输出如下结果（其中两个矩阵分别表示 net1 的两个后继）：

```
[[[ 0 -1 -1  0 -1 -1]
  [-1  0  1  0  1 -1]
  [-1  1  0  0  1 -1]
  [ 0  0  0  0  0  0]
  [-1  1  1  0  0 -1]
  [-1 -1 -1  0 -1  0]]
 [[ 0 -1  0  0 -1  1]
  [-1  0  1  0  1 -1]
  [ 0  1  0  0  1 -1]
  [ 0  0  0  0  0  0]
  [-1  1  1  0  0 -1]
  [ 1 -1 -1  0 -1  0]]]
```

函数 isSucc() 可用来检测一个网络是否是另一网络的后继。如果存在时间线 $N_0 \rightsquigarrow N_1 \rightsquigarrow \cdots \rightsquigarrow N_n$ 使得 $N = N_0$ 且 $N' = N_n$，则称网络 N' 是 N 的 n 步后继。执行命令 isSucc(net1,net2,num)，如果 net2 是 net1 的 num 步后继，则返回 True，否则返回 False。

定义如下网络：

```
net2a = netGen(net1,1,(0,5))
net2b = netGen(net1,-1,(0,2))
net3 = netGen(net2a,-1,(0,2))
netx = netGen(net1,1,(4,5))
```

然后使用检测它们中的某些是否为否另外一些的后继：

```
print(isSucc(net1,net2a,1))
print(isSucc(net1,net2b,1))
print(isSucc(net2a,net3,1))
print(isSucc(net2b,net3,1))
print(isSucc(net1,net3,2))
print(isSucc(net1,netx,1))
```

得到的结果分别为 True、True、True、True、True 和 False。

4.2 语法检测

函数 formChk(n,string) 检验一个字符串是否为 $\mathcal{L}^n$ 的公式，即满足语法规则且只包含 0 到 $n-1$ 主体的字符串。如果满足，则返回值为 True 。若否，则为 False。此外，这个功能也被用在装饰器 argsChk() 中，该装饰器可保证其他函数输入的公式始终符合标准。

例如，运行以下代码得到的输出

```
print(formChk(6,'EU(stable,~P(1,4))'))
print(formChk(2,'AU(stable,~/\(P(1,0),N(0,1)))'))
print(formChk(6,'AX(stable,~P(1,4))'))
print(formChk(6,'sssss,~P(1,4)'))
```

返回的值分别为 True、True、False 以及 False。

4.3 模型检测、有效性检测和可满足性检测的实现

模型检测、有效性检测和可满足性检测分别以函数 mc()、sat() 和 valid() 加以实现。下面给出具体说明。

任给网络 N（由 net 表示）和公式 φ（由 formula 表示），如果 $N \models \varphi$，则命令 mc(net,formula) 返回 True ，否则返回 False。

为方便输入网络，提供了函数 desc() 用以生成网络的描述（描述的定义见第 3 节）。实际输出的是部分的网络描述（矩阵），即仅仅是矩阵对角线右上方的那些连带。这样做并不会丢失信息，因为对于网络这样的无向而言，对角线两边完全对称。例如，print(desc(net1)) 输出如下：

```
/\(N(0,1),/\(O(0,2),/\(O(0,3),/\(N(0,4),/\(N(0,5),/\(P(1,2),
/\(O(1,3),/\(P(1,4),/\(N(1,5),/\(O(2,3),/\(P(2,4),/\(N(2,5),
/\(O(3,4),/\(O(3,5),N(4,5))))))))))))))
```

上面使用的是前缀表达式，改用中缀表示并去除多余括号，即如下公式：

$$\begin{aligned}\mathrm{N}(0,1)\wedge\mathrm{O}(0,2)\wedge\mathrm{O}(0,3)\wedge\mathrm{N}(0,4)\wedge\mathrm{N}(0,5)\wedge\\ \mathrm{P}(1,2)\wedge\mathrm{O}(1,3)\wedge\mathrm{P}(1,4)\wedge\mathrm{N}(1,5)\wedge\\ \mathrm{N}(2,3)\wedge\mathrm{P}(2,4)\wedge\mathrm{N}(2,5)\wedge\\ \mathrm{O}(3,4)\wedge\mathrm{O}(3,5)\wedge\\ \mathrm{N}(4,5)\end{aligned}$$

不难验证，`desc(net1)` 在且只在网络 `net1` 中为真。

任给公式 φ，命令 $\mathtt{sat}(n,\varphi)$ 返回 `True`，如果 φ 是 n 可满足的，即存在 n 节点网络 N 使得 $N\models\varphi$；否则返回 `False`。命令 $\mathtt{valid}(n,\varphi)$ 返回 `True`，如果 φ 是 n 有效的，即所有 n 节点网络 N 都满足 $N\models\varphi$；否则返回 `False`。

目前还未对可满足性和有效性检测工具进行算法优化，只是通过测试所有网络将两类检测问题归约为模型检测问题，因此大型网络的结果输出可能会比较慢。

函数 `satFind(n,formula,mode='normal')` 在默认情况（`'normal'` 模式）下，会查找满足给定 $\mathcal{L}^n$ 公式的所有网络（可能非常耗时）。如果找不到，则返回 `False`。如果设置 `mode='min'`，则输出满足给定公式的最小网络（如果不可满足则返回 `False`）。例如，`print(satFind(5,'AXP(1,2)',True))` 的输出如右图所示。

```
[[0 1 1]
 [1 0 0]
 [1 0 0]]
```

5 结语

本文给出了敌友逻辑 （[10]）的公理系统和程序实现。公理系统包含稳定公理（SF）和明日公理（TF）以及依赖网络描述的规则 Tm1 和 Tm2。这些公理和规则是纯语形的，但很大程度上复制了网络更新的形式语义定义。我们不确定是否可以在不使用网络描述的前提下对敌友逻辑进行公理化，但我们给出的公理系统绝非毫无意义。它是对 CTL 公理系统的保守扩充，至少表明 CTL 的公理和规则对于本文引入的基于网络的形式语义仍然是有效的。

目前的程序实现严格地依照语义定义来进行。这也就意味着程序可以进行优化。例如，如果遇到形如 $\mathrm{AU}(\varphi,\psi)\vee{\sim}\mathrm{AU}(\varphi,\psi)$ 这种可以通过命题逻辑简单判定的公式，程序中仍然需要（两次）检测 $\mathrm{AU}(\varphi,\psi)$ 是否为真。又如，可满足性检测的程序算法是从最小的网络逐一开始检测直到找到满足的网络，但通常来说，还有更好的策略去找到满足条件的网络。优化模型检测算法的方法有很多，这是从事定理证明等相关领域研究者经常面临的任务。本文的重点是给出行之有效的程序实现，而进一步的优化则是未来的工作方向。

网络中边的符号目前仅允许 $\{-1,0,1\}$ 中的，对此进行推广是一件自然而然

的事情。如果将符号的数字理解为概率，那么恰当的符号范围可能是常用于表示概率的实数区间 $[-1, 1]$。在这种情况下，一对主体的概率或许可以视为其共识或分歧符号的乘积。如果将符号理解为关系的强度，则使用自然数、有理数或实数都是可行的。关系的强度可以是共识或分歧中的最小值，或是它们的和。在结构平衡理论中增加关系强度的刻画早在 （[12]）中就已经出现，这些将留作未来的任务。

使用图形表示网络可以使模型检测工具更加实用，我们打算在未来加以实现。除此以外，敌友逻辑适用于不同领域的版本也具有研究价值，我们将探讨允许网络更新算子、与博弈论结合等研究思路。

参考文献

[1] D. Cartwright and F. Harary, 1956, “Structure balance: A generalization of Heider’s theory”, *Psychological Review*, **63(5)**: 277–293.

[2] E. M. Clarke and E. A. Emerson, 1981, “Design and synthesis of synchronization skeletons using branching time temporal logic”, in D. Kozen (ed.), *Logics of Programs*, pp. 52–71, New York: Springer.

[3] E. Emerson and E. M. Clarke, 1982, “Using branching time temporal logic to synthesize synchronization skeletons”, *Science of Computer Programming*, **2(3)**: 241–266.

[4] R. Goldblatt, 1992, *Logics of Time and Computation*, Stanford: CSLI.

[5] F. Harary, 1953, “On the notion of balance of a signed graph”, *Michigan Math. J.* **2(2)**: 143–146.

[6] F. Harary, 1955, “On local balance and N-balance in signed graphs”, *Michigan Math. J.* **3(1)**: 37–41.

[7] F. Harary, R. Z. Norman and D. Cartwright, 1965, *Structural Models: An Introduction to the Theory of Directed Graphs*, New York: John Wiley & Sons Inc.

[8] F. Heider, 1944, “Social perception and phenomenal causality”, *Psychological Review*, **51(6)**: 358–374.

[9] F. Heider, 1946, “Attitudes and cognitive organization”, *Journal of Psychology*, **21(1)**: 107–112.

[10] W. van der Hoek, L. B. Kuijer and Y. N. Wáng, 2018, “A logic of allies and enemies”, *Thirteenth Conference on Logic and the Foundations of Game and Decision Theory (LOFT)*.

[11] T. Jech, 2011, *Set Theory*, Berlin: Springer.

[12] J. O. Morrissette, 1958, “An experimental study of the theory of structural balance”, *Human Relations*, **11(3)**: 239–254.

（责任编辑：罗心澄）

Dynamic Social Network Modeling

—Axiomatizing and Implementing the Logic of Allies and Enemies

Yì N. Wáng Xiling Luo

Abstract

The Logic of Allies and Enemies (LAE) (van der Hoek, *et al.*, 2018) models the dynamics of social networks from the viewpoint of structural balance theory. A network is stable if all pairs of agents have no reason to change their relationship, and an unstable network evolves over time until eventually reaching a stable state. LAE is a CTL-based temporal logic with every timeline modeling evolution of networks. We axiomatize this logic in the first part of the paper. The model and validity checking problems for LAE is known to be PSPACE-complete. We introduce an LAE model checker in the second part.

Yì N. Wáng Department of Philosophy, Zhejiang University
ynw@xixilogic.org
Xiling Luo Department of Philosophy, Zhejiang University
xiling@zju.edu.cn