arXiv:2602.11203v1 [cs.LO] 6 Feb 2026

系统与偏序运行的组合性

Peter Fettke[1,2][0000-0002-0624-4431] 和
Wolfgang Reisig[3][0000-0002-7026-2810]

[1] 德国人工智能研究中心(DFKI)，萨尔布吕肯，德国 peter.fettke@dfki.de[2] 萨尔兰大学，萨尔布吕肯，德国[3] 柏林洪堡大学，德国柏林 reisig@informatik.hu-berlin.de

摘要。 20 世纪 70 年代末，C.A. Petri 引入了偏序事件发生（运行）这一概念，当时称为 *processes*，并将其作为描述分布式系统各自演化过程的恰当模型。本文提出一个统一框架，用于处理 Petri 网及其运行，特别是对它们进行组合与分解。我们证明，对于网 M 和N，组合网 $M \bullet N$ 的运行集合等于 M 和 N的运行的组合。

关键词： 组合性 · 建模理论 · 离散系统 · 行为建模 · 组合演算 · 佩特里网

1 动机

20世纪70年代末，C.A. Petri提出了偏序事件发生（当时称为过程），认为这是描述分布式系统个体演化的恰当模型。由此产生的方法被称为“偏序语义”，以区别于基于发射序列的“交错语义”。尽管（业务）流程管理的重要性如今已广为人知 [5,6,20]，但Petri所理解的过程概念从未得到广泛认可。我们认为，这其中存在概念和技术上的障碍。首先，Petri的过程并非按时间顺序，而是按因果关系对事件发生进行排序。这对受传统教育的计算机科学家来说相当陌生。自动机理论、编程语言语义、分布式系统行为、时序逻辑及其他理论，都会沿时间尺度组织事件发生。这是一种强烈且理想化的假设，忽略了在时间上相近的独立事件所带来的测量问题。虽然这一假设适用于许多系统模型，却可能妨碍人们理解许多现象。其次，交错语义只能处理状态与步骤序列，而过程在技术上更为复杂，尤其是在组合与精化方面。

系统模型及其行为的一个基本属性是组合性：系统的组合必须由其行为上的组合算子来体现

。交错语义借助不太优雅的“交错”算子，在一定程度上满足了这一要求。本文提出一个关于偏序语义组合性的通用框架。

一般而言，组合性原则假设存在两个集合 A 和B，每个集合都带有内部算子：$\bullet_A$： $A \times A \to A$ 和 $\bullet_B$： $B \times B \to B$，以及一个映射 f： $A \to B$。组合性意味着，对于所有 a，$b \in A$，都有 $f(a \bullet_A b) = f(a) \bullet_B f(b)$。例如，在形式语言理论中，自动机和形式语言具有组合性：要组合两个有限自动机 M和 N，将 M的终态与 N的初态合并，从而得到自动机 $M \bullet N$。要组合两个语言 A 和 B，将 v 中的每个词与 A 中的每个词 w 在 B中连接，生成词 vw。自动机和形式语言中的组合性因此表示，自动机 $M \bullet N$的语言 $L(M \bullet N)$ 等于语言 M 和 N的组合 $L(M) \bullet L(N)$。从技术角度来说，对两个组合算子使用相同的符号：

$$L(M \bullet N) = L(M) \bullet L(N). \tag{1}$$

对于库所/变迁网，情况更为复杂：通常假设 Petri 网 N 的每个发生序列都会为 N 的语言 $L(N)$ 贡献一个词。组合 Petri 网的方式有很多。大多数非平凡的组合算子 $\bullet$ 都意味着，对于两个 Petri 网 M 和 N，语言 $L(M \bullet N)$ 包含的词比 $L(M) \bullet L(N)$ 更多。直观地说，在 $M \bullet N$ 中，网 N 可能会在 M 完全完成之前就开始运行。为解决这一问题，人们提出了各种版本的交错算子 [8]。

作为发生序列的一种替代，Petri 提出了偏序运行 [12]。这些运行已得到广泛研究，但仍未取得明确的突破。这主要有两个原因：在概念上，偏序运行以因果顺序取代时间顺序，这要求我们重新审视关于系统行为的经典观念；在技术上，组合和精化等概念对于偏序运行而言有些复杂。

在本文中，我们用灵活且在技术上简单的偏序运行概念取代发生序列，该概念满足方程 1。

本文其余部分安排如下：下一节提供一个入门示例。第 3 节讨论网模块及其组合的概念。第 4 节介绍网模块的步和运行，以及基本步和运行这一特殊情况。第 5 节给出组合定理，并据此确立方程 1。第 6 节简要讨论相关工作。

2 入门示例

为了阐明方程 1 在佩特里网及其偏序运行中的情形，图 1 给出了一个示例，其中包括 $M =$ 面包师系统和 $N =$ 供应商系统。

对于方程 1 的左侧，图 2 展示了由图1中的两个系统组成的全局系统。

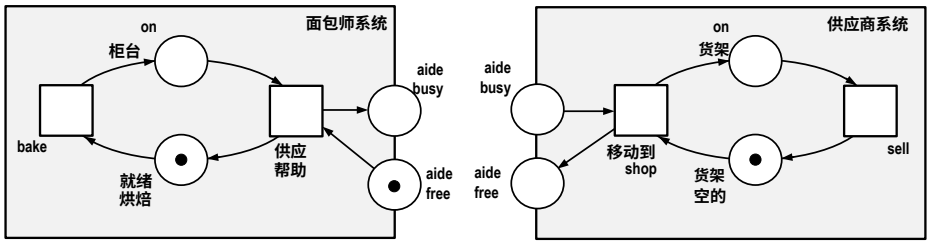


图 1。 面包师系统和供应商系统

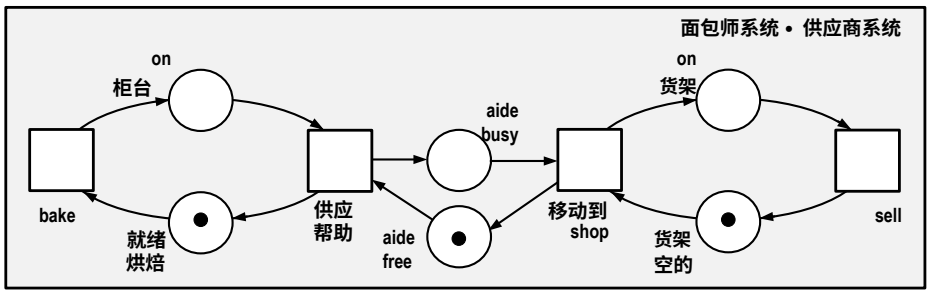


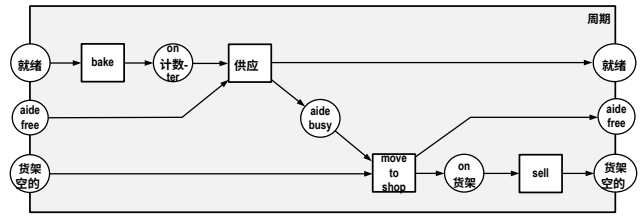
图 2。 由面包师系统和供应商系统组合而成的全局系统

图 3 展示了全局系统的两种运行，该系统由供应商系统和面包师系统组成。图 3(a)（称为周期）展开了一个包含烘焙、供应、移动和出售面包的周期。图 3(b) 展示了两个相互交织的此类周期。直观来说，一次运行会将每个独立的事件发生分别作为一个单独的个体来处理。例如，在图 3 中，我们可以区分烘焙的第一次发生（此前没有任何事件发生），与烘焙的第二次发生（此前发生过供应 的第一次发生，以及烘焙的第一次发生）。因此，用技术术语来说，一次运行中的一系列弧永远不会构成周期。并且，没有任何库所拥有多于一条输入弧或一条输出弧。与系统模块一样，运行也可以组合。例如，图 3(b) 中的运行 R_2 正是图 3(a) 中运行 R_1 的两个实例的组合。

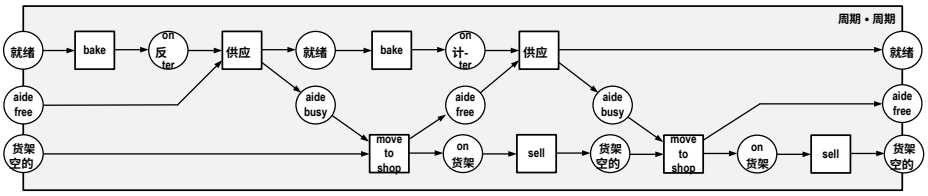
转到方程 1 的右边，我们考察图 1 中面包师系统的运行和供应商系统的运行。图 4 展示了这些运行。上方的运行展示了面包师系统两个周期的组合；下方的运行展示了供应商系统两个运行的组合。请注意，这两个运行的接口都包含标签相同的元素（助手忙碌）。

本节示例的核心要点和动机，现在是观察到：由两个周期组合而成的全局系统运行（如图 3(b) 所示），与图 4 中两个运行的组合相一致！这表明方程 1 对面包师系统和供应商系统均成立。

图 1 至图 4 中的系统和运行，其接口上都带有库所，如各自外围方框边缘所示。接口还可以



（a）运行 R1：全局系统的一个周期



（b）运行 R2：图 2 中全局系统两个周期的组合。

图 3。 全局系统的运行

包含变迁，如图 5 中的系统和图 6 中的运行所示。例如，最上方的运行取货-供货的运行在其右接口上有两次向助手供应事件。第二个运行供应-移动的运行在其左接口上有两次向助手供应事件，在其右接口上有两次移动到商店事件。最后，运行移动-销售的运行在其右接口上有两次移动到商店事件。

图 6 中三个运行的组合再次得到图 3(b) 中的运行。

3 网模块及其组合

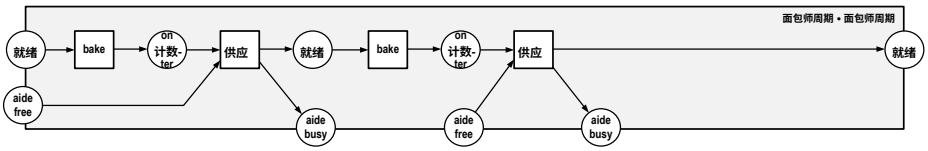
本节乃至全文的核心概念是网模块：一种带有接口的网，这些接口使模块能够与其他模块连接。传统的库所/变迁网以及分布式运行都可以转换为模块。

为保持内容自成一体，我们回顾一些关于集合的序和标记的形式化基础。

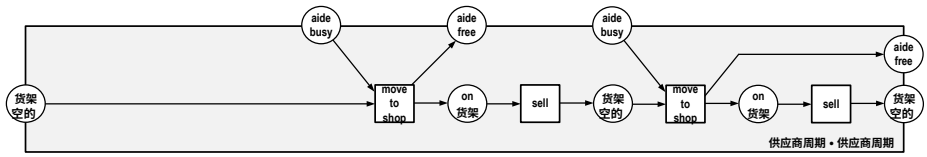
定义 1（有序标记集，度）。

1. 如果假定存在一个全序，则集合 A 是 有序的 。*2.* 如果假定存在一个标签集 Λ 和一个映射 $\lambda : A \to \Lambda$，则集合 A 是 带标记的 。*3.* 有序标记集 A 中元素 a 的 度，是 A 中标签与 a 相同且小于 a 的元素数量。

接下来，我们考虑有序标记集 $A = \{a, b, c, d\}$，按字母顺序排列，其标记如下：



(a) 面包师系统的两个周期



(b) 供应商系统的两个周期

图 4. 面包师系统和供应商系统的运行

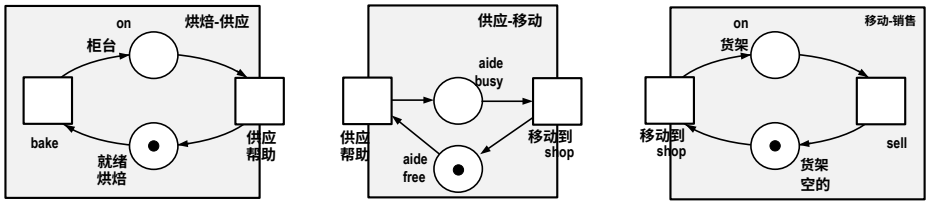


图5。 三个系统模块

$$\lambda(a) = \mathit{ready}, \lambda(b) = \lambda(d) = \mathit{aide\ busy} \text{ and } \lambda(c) = \mathit{aide\ free}. \tag{2}$$

d 的度为 2； A 中所有其他元素的度均为 1。基于这些概念，接口的定义如下：

定义2（接口、匹配伙伴、无匹配）。

1. 接口 是一个有限的有序标记集。*2.* 对于两个接口 A 和 B，如果元素 $a \in A$ 和 $b \in B$ 的标签相同且度相同，则它们分别是 A 和 B 的匹配伙伴 。$\mathrm{matches}(A, B)$ 表示 A 和 B 的所有匹配伙伴组成的集合。*3.* $\mathrm{matchfree}(A, B)$ 包含 $a \in A$ 中没有匹配伙伴的所有元素。*4.* $\mathrm{matchfree}(B, A)$ 包含 A 中没有匹配伙伴的所有元素。

令 B 为有序标记集 $B = \{e, f, g, h\}$，按字母顺序排列，其标记如下：

$$\lambda(e) = \mathit{shelf\ empty}, \lambda(f) = \lambda(h) = \mathit{aide\ busy} \text{ and } \lambda(g) = \mathit{aide\ free}. \tag{3}$$

h 的度为2； B 的所有其他元素的度均为1。此外，方程2和3中集合 A 和 B的匹配伙伴分别为 b 和

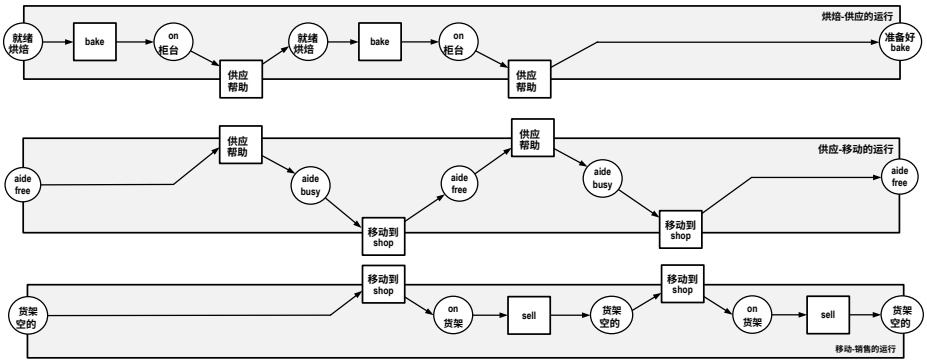


图6. 图5中三个系统模块各自的两个周期

f、 c和 g，以及 d和 h。无匹配$(A,\ B)$包含 a，而无匹配$(B,\ A)$包含 e。

集合匹配(A, B)、无匹配(A, B)和无匹配(B, A)分别继承 A 和 B的顺序与标记，并且它们自身也可以视为接口。

Petri网的网图结构按通常方式定义：

定义3（网图）。设 P 和 T 为有限且不相交的集合。设 E 为 $P \cup T$上的一个关系，使得对于每个元组$(x, y) \in E$，要么满足 $x \in P$和 $y \in T$，要么满足 $x \in T$和 $y \in P$。那么 $M = (P, T; E)$ 是一个网图。 P、T和 E 的元素分别称为库所、变迁和弧。

通常，对于网图，我们假设库所和变迁分别具有独立的标签集 Λ_P 和 Λ_T。现在我们可以将网模块定义为带有左接口和右接口的网图：

定义 4（网模块、左接口和右接口、模块内部）。令 $M = (P, T; E)$ 为一个网图。令 ${}^*M, M^* \subseteq P \cup T$为接口。那么 M连同 ${}^*M$和 M^*构成一个网模块。 ${}^*M$和 M^*分别是 M的左接口和右接口。 $P \cup T \setminus {}^*M \cup M^*$是 M的内部。

记号出于技术原因，我们有时使用空网模块$(\emptyset, \emptyset, \emptyset)$，也记作 $[\emptyset]$。

图形约定在图示表示中，网图通常绘制在 Petri 网区域内，并使用圆圈、方块和箭头。对于网模块，还采用了若干额外的图形约定，以最直观的方式直观呈现模块的组合。

按照惯例，网模块 M 被框在一个方框中，左接口和右接口的元素分别置于方框的左、右边缘。接口元素的身份通常会省略，接口元素按垂直方向排列，并在每个元素上标注其标签。例如，在图 1 中，面包师系统模块的右接口和

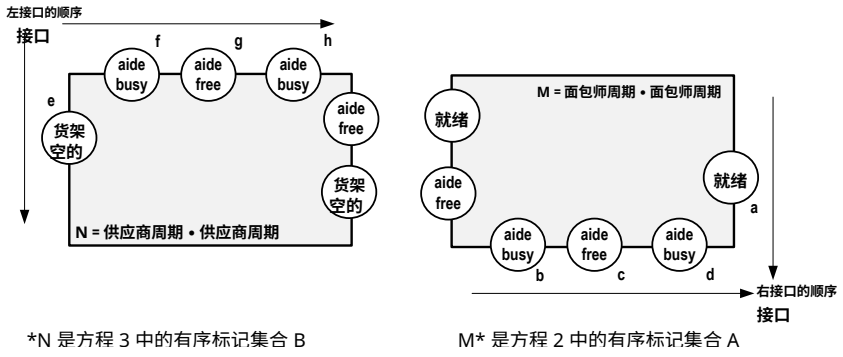


图 7. 图 4 中模块的抽象版本

供应商系统模块的左接口都包含两个库所。第一个库所的标签是“助手忙碌”，第二个库所的标签是“aide free”。面包师系统模块的左接口和供应商系统模块的右接口均为空。作为第二个示例，图 3 展示了两个模块，它们的左、右接口标签均相同。

通常，将左接口元素画在外包围框的上边缘，将右接口元素画在下边缘会很有帮助，如图 4 所示。两种情况下，顺序都从左向右，并延续左右边缘上元素自上而下的顺序。图 7 展示了方程 2 和 3 中带标签且有序的集合所采用的这一约定。巧合的是，图 7 展示了图 4 中模块的抽象版本。

现在，我们准备好介绍本文的核心思想：网模块的组合。

定义 5（模块的组合）。设 M 和 N 是网模块。M与 N的组合是网模块 $M \bullet N$，其中

– $M \bullet N$ 的内部为 $interior(M) \cup interior(N) \cup matches(M^*, {}^*N)$；– 左接口 $^*(M \bullet N)$ 为 $^*M \cup matchfree(^*N, M^*)$。$^*M$ 的元素排在 $matchfree(^*N, M^*)$ 的元素之前。– 右接口 $(M \bullet N)^*$ 为 $N^* \cup matchfree(M^*, {}^*N)$。$N^*$ 的元素排在 $matchfree(M^*, {}^*N)$ 的元素之前。– $M \bullet N$ 继承 M 和 N 的弧；每个匹配 (a, b) 继承以 a 或 b 为起点或终点的弧。

图 8 展示了这个定义。接口元素的标签为 a、.....、e。模块 M和 N的内部未显示。$M \bullet N$的内部由 M的内部、N的内部，以及 M^*和 $^*N$中带有 b标签的元素组成的元组构成。显然，M和 N的每个元素要么是 $M \bullet N$的元素，要么会促成 M^*和 $^*N$的匹配，而该匹配本身又是 $M \bullet N$的一个元素。图 2 中的整体系统是图 1 中的面包师系统与供应商系统的组合。图 3(b) 展示了两个

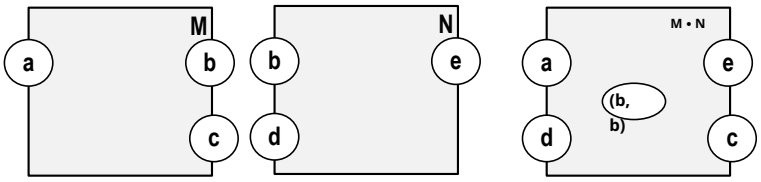


图 8. 组合操作示意图

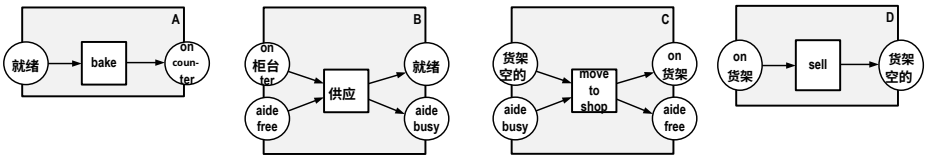


图 9. 图中网模块的步 . 2

图 3(a) 实例的组合。此外，图 3(b) 也可由图 4(a) 和图 4(b) 组合得到。

显然，对于空网模块 $[\emptyset]$ 有： $M \bullet [\emptyset] = [\emptyset] \bullet M = M$。

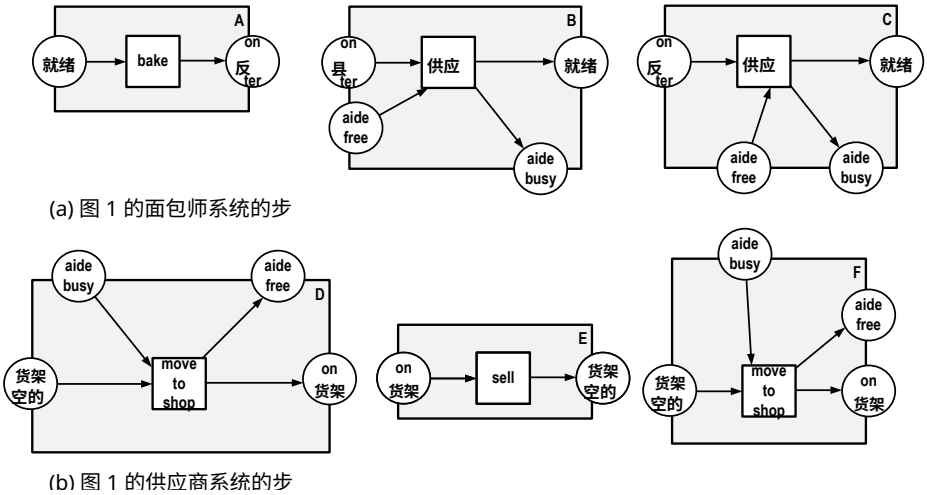
4 网模块的运行

网模块 M 的一次运行代表其众多演化行为之一。一次运行由步组成。变迁 t 在 M 中的一步表示 t的发生：被舍弃的局部状态和进入的局部状态。从技术上讲， t 的一步 S 是一个只有单个变迁 u 、且该变迁的标签为“t”的带标记的网模块。 S中 u 的周围库所及其标签，反映了 M中 t 的周围库所的相应模式。图 9 展示了图 2 中网模块的各步。这些步的结构相当规整：被舍弃的状态构成左接口，进入的状态构成右接口。

定义 *6*（网模块的一步）。设 M和 $A =$ （Q、 u、 G）为网模块。若且唯若（λ（p） λ（u））是 M的一条弧，则（p， u）是 A的一条弧；若且唯若（λ（u） λ（p））是 M的一条弧，则（u， p）是 A的一条弧。那么， A是 M的一步。

请注意，此定义没有以任何方式限制一步的接口。 A 的任何元素子集都可以充当 A的接口。因此，情况并不总像图 9 所示的那样简单。在图 10 中，步 C的右接口上包含带有标签 $aide\ free$ 的进入状态。步 D的右接口上包含带有标签 $aide\ free$ 的退出库所。图 11 展示了接口内的变迁这一情形。

事情并不总是那么简单，如图 10 所示。图 11 展示了接口中变迁的情形。



(a) 图 1 的面包师系统的步

(b) 图 1 的供应商系统的步

图 10. 面包师系统与供应商系统的运行的步

并非每个具有无分支位置的无环模块都能由步组合而成：此类模块的内部可能需要一种步序，而这种步序与接口元素上的步所生成的顺序相冲突。图 12 展示了一个例子。因此，我们将网模块的运行归纳定义为步的组合。

网模块 M 的运行现在将被正式定义为由 M 的步组成。步和运行也都是网模块；因此，它们按定义 5 中所述进行组合。根据设计，运行是无环的，其库所不会分支，即每个库所至多有一条弧起始于此，且至多有一条弧终止于此。然而，两个运行的组合不一定构成运行：结果可能包含一个周期。即使是步的组合也可能产生周期，如图 14 所示。

定义 7（runs(M)）。令 M 为一个网模块。M 的运行集合 $\mathrm{runs}(M)$ 归纳定义如下：

1. 空网模块 $[\emptyset]$ 是 M 的一个运行。*2.* 令 R 为 M 的一个运行，并令 A 为 N 的一个步，且 $R \bullet A$ 是偏序的。则 $R \bullet A$ 是 M 的一个运行。

例如，图 3 中的两个运行实际上都是图 2 所示全局系统的运行。上方的运行 R_1 可以直接由图 8 中的步组成，即 $R_1 = A \bullet B \bullet C \bullet D$。下方的运行 R_2 则可以写作 $R_2 = R_1 \bullet R_1$。此外，图 4 中上方的运行模块对应于图 1 所示面包师系统的一个运行。它可以由图 10(a) 中的步组装而成，即 $A \bullet B \bullet A \bullet B$。同样，图 4 中下方的运行模块是图 1 中供应商系统的一个运行，由图 10(b) 中的步组成，即 $D \bullet E \bullet D \bullet E$。最后，图 6 中的三个运行由图 11 中的步构成，分别为 $A \bullet B \bullet A \bullet B$、$C \bullet D \bullet C \bullet D$ 和 $E \bullet F \bullet E \bullet F$。

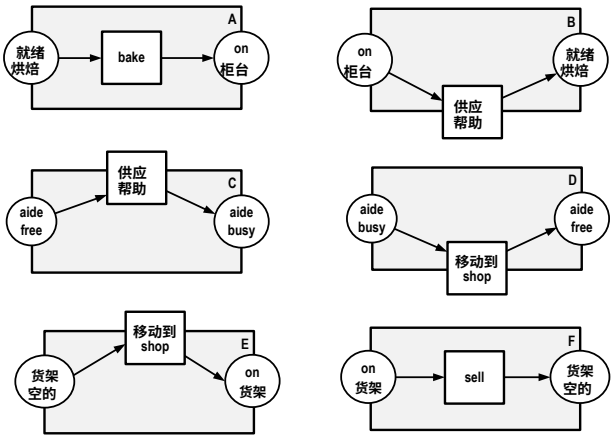


图 11. 图5中三个系统模块的步

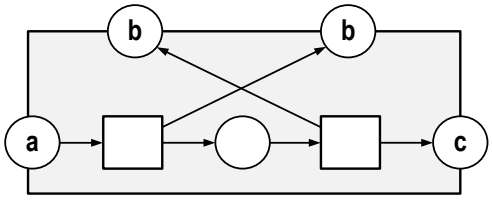


图 12. 具有未分支库所的偏序网模块

到目前为止，所有系统示例都表示为具有无分支位置的网模块。尤其是图 1、图 2 和图 5 中的网模块。定义 7 中对运行的定义并不要求这一限制。对于具有分支位置的网模块，步和运行也都有很好的定义。图 13 展示了一个示例：在图 2 的系统中增加了第二个供应商。图 2 中全局系统的所有运行也都是图 13 所示系统的运行。

按照构造，运行是无环的，其库所不会分支，也就是说，每个库所至多有一条弧起始于此，至多有一条弧终止于此。然而，两个运行的组合不一定是运行：结果可能包含一个周期。图 14 展示了一个示例。

在许多建模上下文中，步的结构相当规则：入弧起始于左接口；出弧终止于右接口。图 8 展示了这些基本步的典型示例。此类步的组合随后会生成一个基本运行。

定义 8（基本运行）。运行 R 是基本的，当且仅当 ${}^*R$ 由 R中所有没有入弧的库所组成，并且 R^* 由 R 中所有没有出弧的库所组成。

图 3 展示了典型的基本运行。基本步和运行的范围在组合下是封闭的：

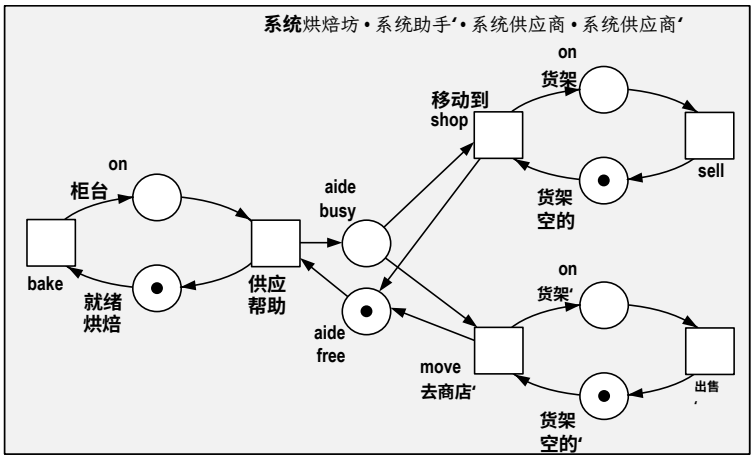


图 13. 带有第二个卖家的全局系统 or 图 2

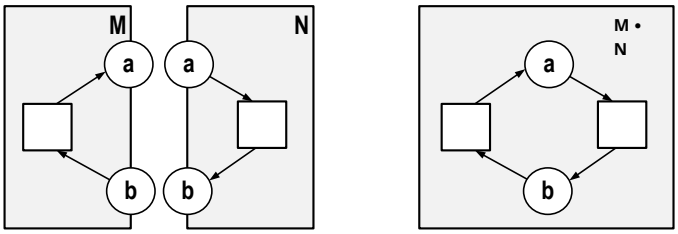


图 14. 两个技术步骤及其组合

定理 1. 基本运行 R 与 S 的组合 $R \bullet S$ 是一个基本运行。

证明。 设 R 和 S 为基本运行。根据定义8，我们需要证明：

– $R \bullet S$ 是一个运行，即不包含圈，– $^*(R \bullet S)$ 由 $R \bullet S$ 中在 $R \bullet S$ 里没有入弧的所有库所组成，– $(R \bullet S)^*$ 由 $R \bullet S$ 中在 $R \bullet S$ 里没有出弧的所有库所组成。

图15概述了这一情况。

为证明 (i)，注意到 R^* 中没有任何元素在 R 中具有出弧，并且 $^*S$ 中没有任何元素在 S 中具有入弧，因为 R 和 S 是基本运行。因此，$R \bullet S$ 不包含从 S 到 R 的弧。因此， $R \bullet S$ 不包含任何圈，因为 R 和 S 不包含任何圈。

为证明 (ii)，注意到 $^*S$ 中对 $^*R$ 有贡献的元素，在 S 中没有入弧，因为 S 是基本运行。因此， $^*S$ 中对 $^*(R \bullet S)$ 有贡献的元素，在 $R \bullet S$ 中没有入弧。

为证明 (iii)，注意到，在 S 中， R^* 中对 S^* 有贡献的元素都没有出弧，因为 R 是一个基本运行。因此，在 $R \bullet S$ 中， R^* 中对 $(R \bullet S)^*$ 有贡献的元素都没有入弧。

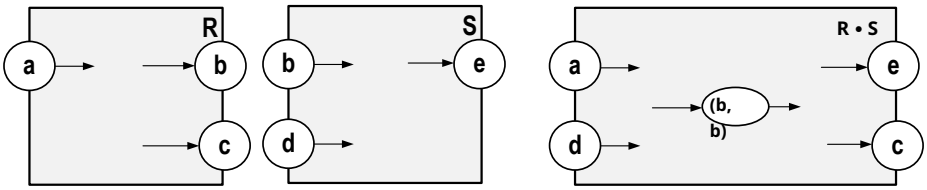


图15. 定理 1 的证明概要

5 组合定理

定义 5 定义了模块的组合，因此引言中方程 1 的左侧是良定义的。右侧则需要定义运行集合的组合。为此，我们按通常方式将运行的组合提升到运行集合A、B上：$A \bullet B$由 A与B的所有组合运行构成。

定义 9. 设 A 和 B 是两个运行集。那么 $A \bullet B = \{a \bullet b \mid a \in A$ 和 $b \in B$，且 $a \bullet b$ 是一个运行$\}$。

定理 2。 对于任意两个系统网 M 和 N，有：

$$\mathrm{runs}(M \bullet N) = \mathrm{runs}(M) \bullet \mathrm{runs}(N).$$

证明。 首先，我们证明：运行$(M \bullet N) \subseteq$运行$(M) \bullet$运行(N)。

需要证明：对于 $M \bullet N$的每个运行 R，都存在 M的运行 R_M和N的运行 R_N，使得 $R = R_M \bullet R_N$。

因此，设 $R = (P, T; F)$ 是 $M \bullet N$的一个运行，其标记函数为 λ。我们对 $|T|$进行归纳，以证明该命题。

1. 归纳基础：设 $|T| = \emptyset$。则 R 是空模块 $[\emptyset]$。 $[\emptyset]$ 也是 M的一个运行，并且是N的一个运行。此外，显然有：$[\emptyset] \bullet [\emptyset] = [\emptyset]$。 2. 归纳步骤：设 $|T| = k$，其中 $k > 0$。则 R 可以写成 $R = R_1 \bullet \ldots \bullet R_k$，其步为 R_1、.....、R_k，属于 $M \bullet N$（根据定义 7）。由归纳假设，存在 M的运行 R_M和 N的运行 R_N，使得 $R_1 \bullet \ldots \bullet R_{k-1} = R_M \bullet R_N$。设 $R_k = (P, \{t\}; E)$，并设 $\lambda(t) = u$。那么 u要么是 M的一个变迁，要么是 N的一个变迁，要么是 M 的一个变迁u_M 与 N 的一个变迁 u_N 的匹配（根据定义 5）。因此，我们分三种情况讨论：

   – 情形 1： u 是 M的一个变迁。因此， R_k 是 M 的一个步（根据定义 6）。因此，R_k 是 $M \bullet N$的一个步（根据定义 5）。因此， $R_M \bullet R_k$是 M的一个运行。因此，$R_M \bullet R_k \bullet R_N$是 $M \bullet N$的一个运行（根据定义 7）。– 情形 2： u 是 N的一个变迁。因此， R_k 是 N 的一个步（根据定义 6）。因此，R_k是 $M \bullet N$的一个步（根据定义 5）。因此， $R_N \bullet R_k$是 M的一个运行。因此，$R_M \bullet R_N \bullet R_k$是 $M \bullet N$的一个运行（根据定义 7）。

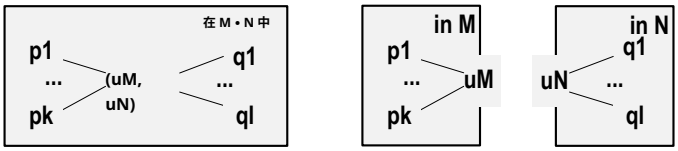


图 16. 步骤 3 的组成部分

–情形 3： u是一个形如 $u = (u_M, u_N)$的变迁，其中 u_M是M 的一个变迁，u_N 是 N的一个变迁，并且 u_M 和 u_N 分别是 M^*和 $^*N$的匹配伙伴。因此，步 $R_k = (P, \{t\};\ E)$可划分为 M的一个步$S_M = (P_M, \{t_M\}, E_M)$和 N的一个步 $S_N = (P_N, \{t_N\};\ E_N)$，使得 $R_k = S_M \bullet S_N$。因此， $R_M \bullet R_N \bullet S_M \bullet S_N \bullet R_k$ 是 $M \bullet N$的一个运行（根据定义 7）。图 16 概述了这一情形。

要证明 运行(M) •运行(N) ⊆运行($M \bullet N$)，令 R为 运行(M) •运行(N) 的一个运行。我们需要证明 R是 $M \bullet N$ 的一个运行。

R可写作 $R = A_1 \bullet \ldots \bullet A_l$，其中 M 的步为 $A_1, \ldots, A_k$，而 N 的步为$A_{k+1}, \ldots, A_l$，其中 $1 \leq k \leq l$。那么 $A_1, \ldots, A_l$是 $M \bullet N$ 的步（根据定义 6）。因此， R 是 $M \bullet N$ 的一个运行（根据定义 7）。

6 相关工作

偏序语义通常不被视为交错语义的一种根本性、概念性的替代方案，而是被视为加快分析算法的一种技术手段。具体而言，某些偏序语义旨在缓解分布式系统中的“状态爆炸问题”。例如，流程挖掘中的一个典型例子是 [17]。

Petri网的组合历史悠久，相关研究成果众多。这里只提及过去三十年中的一些相关思想： [4]一次组合多个 Petri 网模块，将具有相同标签的库所和变迁合并。文中没有讨论组合的结合律，但显然，这种组合算子并不满足结合律。一项历史悠久且有多种变体的研究方向是 Petri网的代数演算系统，例如盒演算和各种形式的佩特里网代数 [3]。本文汇集了关于无限偏序运行的性质。秉承过程代数的理念，这些演算系统通过使用各种组合算子归纳地定义网的类别，并合并具有相同标签的变迁。组合的结合性大多被视为理所当然，却很少得到明确讨论。 [2] 在范畴框架内定义了“可组合”网 A 与 B 的组合 $A \bullet B$，其中 A 和 B 共享一个网片段 C。研究探讨了组合对分布式运行的影响，但没有讨论组合是否满足结合律。

[10] 提出了一个关于模块化Petri网的通用框架，即具有将网络与其环境进行组合的特征的网。一个模块可能出现在多个实例中。本文未讨论组合的结合性，但对此作了

隐含假设。 [1]是近期一篇关于Petri网组合算子的综述。

文献 [13] 和 [18]讨论了具有双面接口的佩特里网。文中介绍了带边界的佩特里网（PNB），它具有两个组合算子。有趣的是，这两个算子都满足结合律。

文献中也有用于佩特里网以外建模技术的双面接口，不过这些接口往往比本文所示的版本更专门。一个常见的例子是语言 CSP [16]中使用的“管道连接”或“链式连接”算子 $P >> Q$ 。 $>>$ 的结合律无需进一步讨论便被默认成立。

将运行与模块化结合起来的研究寥寥无几。早期且详尽的研究包括 [19] 和 [11]。 [9] 提出了一种组合偏序语义。该组合算子只是将标签相同的接口元素融合在一起。然而，它不满足结合律。

本文所用模块和组合算子的初步构想，源自 [14]中对佩特里网运行的描述。组合算子的结合性已在 [15]中得到证明。详情请参阅 [7]。我们的运行构成一个幺半群，这与形式语言的情形类似。本文的主定理以组合的方式关联系统及其运行。据我们所知，我们的组合算子兼具双面接口、结合组合和组合性，无可匹敌。

参考文献

1. Amparore, E.G., Donatelli, S.：佩特里网组合的来龙去脉。载于：Bernardinello, L., Petrucci, L.（编）《Petri 网与并发的应用与理论——第43届国际会议 PETRI NETS 2022》，挪威卑尔根，2022年6月19日至24日，论文集。计算机科学讲义，第13288卷，第278–299页。Springer（2022） 2. Baldan, P., Corradini, A., Ehrig, H., K¨onig, B.：开放佩特里网：非确定性过程与组合性。载于：Ehrig, H., Heckel, R., Rozenberg, G., Taentzer, G.（编）《图变换：第4届国际会议 ICGT 2008》，英国莱斯特，2008年9月7日至13日，论文集。第257–273页。Springer（2008） 3. Best, E., Devillers, R., Koutny, M.：《佩特里网代数》。Springer Science & Business Media（2001） 4. Christensen, S., Petrucci, L.：迈向有色佩特里网的模块化分析。载于：《佩特里网应用与理论国际会议》。第113–133页。Springer（1993） 5. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.：《业务流程管理基础》。Springer，第2版（2018） 6. Fettke, P., Di Francescomarino, C.：业务流程管理与人工智能。《KI - K¨unstliche Intelligenz》**39**(2), 67–79（2025） 7. Fettke, P., Reisig, W.：《理解数字世界：使用 HERAKLIT 建模》。Springer（2024） 8. Gischer, J.L.：交错语言、佩特里网与上下文有关文法。《Commun. ACM》**24**(9), 597–605（1981）。https://doi.org/10.1145/358746.358767

9. Kindler, E.：Petri 网组件的组合偏序语义。载于：《Petri 网应用与理论 1997：第18届国际会议 ICATPN’97》，法国图卢兹，1997年6月23–27日，论文集18。第235–252页。Springer（1997） 10. Kindler, E., Petrucci, L.：迈向模块化 Petri 网标准：一种形式化。载于：《Petri 网应用与理论：第30届国际会议 PETRI NETS 2009》，法国巴黎，2009年6月22日至26日，论文集30。第43–62页。Springer（2009） 11. Linde-G¨oers, H.G.：《Petri 箱的组合偏序语义》。Shaker（1994） 12. Petri, C.A.：《非顺序过程》，Interner Bericht isf-77-5。Gesellschaft f¨ur Mathematik und Datenverarbeitung（1977） 13. Rathke, J., Soboci´nski, P., Stephens, O.：佩特里网中的组合可达性。载于：《可达性问题：第8届国际研讨会 RP 2014》，英国牛津，2014年9月22日至24日，论文集8。第230–243页。Springer（2014） 14. Reisig, W.：论 Petri 网模式的表达能力。载于：《佩特里网应用与理论国际会议》。第349–364页。Springer（2005） 15. Reisig, W.：具有双侧接口的组件的结合组合。《Acta Informatica》**56**(3), 229–253（2019） 16. Roscoe, A.W.：《理解并发系统》。Springer Science & Business Media（2010） 17. Siddiqui, A., van der Aalst, W.M.P., Schuster, D.：通过 Petri 网展开计算偏序轨迹的对齐。载于：Amparore, E.G., Mikulski, L.（编）《Petri 网与并发的应用与理论——第46届国际会议 PETRI NETS 2025》，法国巴黎，2025年6月22日至27日，论文集。计算机科学讲义，第15714卷，第411–432页。Springer（2025） 18. Soboci´nski, P.：Petri 网交互的表示。载于：《并发理论国际会议》。第554–568页。Springer（2010） 19. Vogeler, W.：《Petri 网的模块化构造与偏序语义》，计算机科学讲义，第625卷。Springer（1992） 20. Weske, M.：《业务流程管理：概念、语言与架构》。Springer 柏林、海德堡，第3版（2019）