文章编号: 1001-0920(2001)03-282-06

基于 ESHLEP-N 的产品开发过程建模、调度与仿真

汪 峥, 严洪森, 刘霞玲, 宋文忠
(东南大学 自动化研究所, 江苏 南京 210096)

摘 要: 为获得可行的产品开发计划, 用扩展随机高级判断 Petri 网建立产品开发过程的模型, 并给出产品开发过程的仿真方法。仿真流程的输入是由数学方法获得的初始产品开发计划, 仿真过程中使用 4 类规则来调度产品设计活动。根据仿真结果考察初始计划的可行性, 并获得实际的产品开发计划。仿真结果表明, 该调度规则和仿真流程是有效而可行的。

关键词: 产品开发过程; 设计活动; Petri 网; 建模; 调度; 仿真
中图分类号: TP 15 **文献标识码**: A

ESHLEP-N Based Modeling, Scheduling and Simulation of Product Development Process

WANG Zheng, YAN Hong-sen, LIU Xia-ling, SONG Wen-zhong
(Research Institute of Automation, Southeast University, Nanjing 210096, China)

Abstract: To obtain a feasible product development plan, the extended stochastic high-level evaluation Petri nets (ESHLEP-N) is employed to model product development process and the method of simulating product development process is presented based on the model. The input of the simulation procedure is the initial product development plan obtained by mathematical methods. In the simulation procedure, four types of rules are used to schedule the design activities. The feasibility of the initial plan is verified and the practical plan is obtained according to the simulation results. The simulation results show that the simulation procedure and the scheduling rules are effective and feasible.

Key words: product development process; design activity; Petri nets; modeling; scheduling; simulation

1 引 言

一个产品项目开发之前, 需要制订可行的产品开发计划。如果忽略产品开发过程中的某些细节, 则可通过数学方法获得初始计划, 而初始计划的可行性需要通过仿真来考察, 并且根据仿真结果修正初始计划, 以获得实际的产品开发计划。本文要解决的问题是: 如何建立产品开发过程的面向仿真的模型, 以及如何对产品开发过程进行调度仿真。

目前已有多种产品开发过程模型, 例如: 描述设计活动之间技术依赖关系的设计结构矩阵模型[1], 描述设计活动之间逻辑关系的 IDEF3 模型[2], 以及基于设计活动网络的项目调度数学模型[3]等。但上述模型都是面向性能分析而不是面向仿真, 它们只体现了产品开发过程的一个或几个方面的特点, 而忽略了其它方面。面向仿真的模型应能较全面地描

收稿日期: 2000-03-18; **修回日期**: 2000-06-06
基金项目: 国家自然科学基金项目(69884001); 江苏省自然科学基金项目(BK95047506)
作者简介: 汪峥(1973—), 男, 江苏南京人, 博士生, 从事并行工程、项目管理等研究; 严洪森(1957—), 男, 浙江江山人, 教授, 博士生导师, 美国工业工程师学会高级会员, 从事生产计划与调度、CMS 等研究。

述产品开发过程的细节特征和不确定性。产品开发过程可看作一个离散事件动态系统, 因而用 Petri 网为其建立面向仿真的模型是合适的。

Petri 网已在许多领域得到广泛的应用, 特别是在制造系统中[4]。综合多种 Petri 网的特点, Yan 等人[5,6]提出了扩展高级判断 Petri 网(EHL EP-N)和扩展随机高级判断 Petri 网(ESHL EP-N), 分别用于对柔性制造系统的控制和仿真。Yan 等[7]还提出另一种形式的 ESHL EP-N, 用于并行工程的过程建模。本文采用这种形式的 ESHL EP-N 为产品开发过程建立面向仿真的模型。

为了对产品开发过程进行仿真, 本文给出 4 类设计活动调度规则, 即活动排序规则、资源分配规则、状态改变规则和仿真终止规则, 并在这些调度规则的基础上给出了仿真流程。最后, 以某汽车传动系统的开发过程为例说明产品开发过程的建模、调度和仿真方法。

2　基于 ESHL EP-N 的产品开发过程建模

一个企业中可能有多个产品开发项目同时进行, 而每个产品开发项目又由多个设计活动组成, 因此产品开发过程可用设计活动网络来描述。在此网络中, 一个设计活动(正常设计或修改设计) 称为可施行的, 当其所有的紧前活动都已完成; 一个设计活动称为不可施行的, 当其紧前活动尚未全部完成。若一个可施行的设计活动获得了必要的资源(人力或设备), 它便可开始进行; 若上游设计活动中发生的错误在下游设计活动中被发现, 正常设计活动就要中断, 以便修改设计; 若所有的设计活动都已结束, 则整个产品开发过程结束。

在产品开发过程中, 人力资源有可能缺席, 设备资源有可能发生故障。因此, 一个设计活动有多种状态: 可施行或不可施行, 正常设计或修改设计, 结束或未结束; 资源也有多种状态: 忙或闲或缺席。这些状态可用 ESHL EP-N 中库所内的令牌来描述, 状态的变化可用 ESHL EP-N 中的变迁来描述[7]。

不失一般性, 假设有两个产品开发项目同时进行, 它们分别由 28 和 24 个设计活动组成, 并且假定共有 20 个人力资源(如设计人员) 和 20 个设备资源(如计算机) 可供使用。根据文献[7] 中 ESHL EP-N 的定义, 可建立上述产品开发过程的 ESHL EP-N 模型(见图 1)。图 1 中各普通库所、决策点和变迁的意义如下:

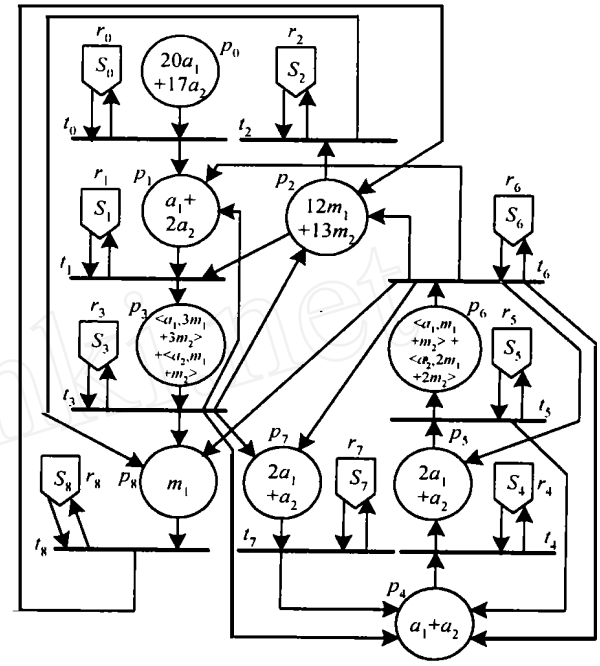

图 1　产品开发过程的 ESHL EP-N 的模型

p_0 是不可施行的正常设计活动库所, p_1 是可施行的正常设计活动库所, p_2 是空闲资源库所, p_3 是正常设计库所; p_4 是不可施行的修改设计活动库所; p_5 是可施行的修改设计活动库所, p_6 是修改设计库所, p_7 是已结束的正常设计活动库所, p_8 是缺席资源库所; r_0～ r_8 是决策点; 决策点内的令牌表示规则集, 即 $S_0, S_1, \cdots, S_8$。变迁 t_0 的触发表示某些正常设计活动成为可施行的; 变迁 t_1 的触发表示正常设计活动开始; 变迁 t_2 的触发表示某些设计人员缺席或某些设备资源发生故障; 变迁 t_3 的触发表示某些正常设计活动结束, 某些正常设计活动的错误被发现, 某些设计人员缺席, 某些设备资源发生故障, 或所有正在进行的正常设计活动需要重新分配资源; 变迁 t_4 的触发表示某些修改设计活动成为可施行的; 变迁 t_5 的触发表示修改设计活动开始; 变迁 t_6 的触发表示某些修改设计活动结束, 某些设计人员缺席, 某些设备资源发生故障, 或所有正在进行的修改设计活动需要重新分配资源; 变迁 t_7 的触发表示某些已结束的正常设计活动的错误被发现; 变迁 t_8 的触发表示缺席的设计人员重新开始工作或发生故障的设备资源修理完毕。

在产品开发过程的 ESHL EP-N 模型中(图 1), 普通库所的所有可能的令牌可表示如下

$$A_p(p_0) = A_p(p_1) = A_p(p_4) =$$

$A_p(p_5) = A_p(p_7) =$
$\{a_{hi}, h = 1, \cdots, g, i = 1, \cdots, s_h\}$
$A_p(p_2) = A_p(p_8) =$
$\{m_{uv}, u = 1, \cdots, q, v = 1, \cdots, r_u\}$
$A_p(p_3) = A_p(p_6) =$
$\left\{\left\langle a_{hi}, \sum_{u=1}^{q}\sum_{v=1}^{r_u} x_{uvhi}m_{uv}\right\rangle, h = 1, \cdots, g, i = 1, \cdots, s_h\right\}$

其中, a_{hi} 为第 h 个产品开发项目的第 i 个设计活动, g 为项目数, s_h 为第 h 个项目的设计活动数, m_{uv} 为第 u 类资源的第 v 个资源, q 为资源种类数, r_u 为可供使用的第 u 类资源数; $\left\langle a_{hi}, \sum_{u=1}^{q}\sum_{v=1}^{r_u} x_{uvhi}m_{uv}\right\rangle$ 表示某些设计人员使用某些设备资源进行正常或修改设计工作, 若 m_{uv} 分配给 a_{hi}, 则 $x_{uvhi} = 1$, 否则 $x_{uvhi} = 0$。

ESHLEP-N 模型中的有色令牌用于观察产品开发过程的状态并进行性能分析。如从有色令牌的分布可知在任意时刻有多少设计活动正处于正常(或修改) 设计状态, 或有多少资源正处于忙(或闲) 状态。图 1 中各普通库所中所有可能的令牌色定义为

$C(A_p(p_0)) = C(A_p(p_1)) = C(A_p(p_4)) =$
$C(A_p(p_5)) = C(A_p(p_7)) = \{a_h, h = 1, \cdots, g\}$
$C(A_p(p_2)) = C(A_p(p_8)) = \{m_u, u = 1, \cdots, q\}$
$C(A_p(p_3)) = C(A_p(p_6)) =$
$\left\{\left\langle a_h, \sum_{u=1}^{q} n_u m_u\right\rangle, h = 1, \cdots, g\right\}$

其中, a_h 为第 h 个项目的一个设计活动, m_u 为第 u 种类型的一个资源, n_u 为第 u 类资源数, $\left\langle a_h, \sum_{u=1}^{q} n_u m_u\right\rangle$ 表示某些设计人员使用某些设备资源进行正常或修改设计工作。为简明起见, 图 1 中仅标出了初始有色令牌。决策点中的令牌(调度规则) 可重复使用, 即一旦某变迁触发, 其相应决策点中的令牌便会移出, 然后又放回此节点。因此在分析 ESHLEP-N 的行为时, 决策点中的令牌可不予考虑。

3 产品开发过程调度规则及仿真流程

ESHLEP-N 模型中决策点内的令牌是设计活动调度规则, 通过调度设计活动来执行产品开发计划。本节将给出 4 类调度规则, 并在此基础上给出仿真流程, 仿真按均匀时间流方式进行。

3.1 调度规则

1) 活动排序规则:

规则 1 若一个不可施行的设计活动成为可施行的或一个可施行的设计活动变成不可施行的, 则所有设计活动都要重新排序。

规则 2 可施行的设计活动可按下列顺序赋予优先级: ① 时差越小, 优先级越高; ② 目标交付期越早, 优先级越高; ③ 随机赋予优先级。可施行的设计活动的优先级各不相同。

规则 3 所有不可施行的设计活动赋予最低优先级。

2) 资源分配规则:

规则 4 若 τ 时刻设计活动优先级或可用资源数不同于 $\tau - 1$ 时刻, 则需要重新分配资源。

规则 5 按设计活动优先级从高到低的顺序确定应分配给各活动的资源数。

规则 6 按设计活动优先级从高到低的顺序给各活动分配资源。当给 a_{hi} 分配第 u 类资源时, 资源按其对 a_{hi} 的优先级从高到低的顺序选取。资源对设计活动的优先级是根据它分配给设计活动的先后确定的, 先分配者优先级高。

3) 状态改变规则: 此类规则的功能是使能并触发 ESHLEP-N 模型中的变迁, 改变设计活动与资源的状态。特别地, 规则 7 和 8 分别给出了正常和修改设计活动已完成工作量和时差的更新方法。

规则 7 在时刻 τ, 对于相应令牌(及有色令牌) 在库所 p_3 中的设计活动 a_{hi}(设其优先级为 θ), 计算 $\mu(\theta, \tau) = \min\{\eta_u(\theta, \tau)/r_{\theta u}, u = 1, \cdots, q\}$。这里, $r_{\theta u}$ 表示具有优先级 θ 的设计活动对第 u 类资源的要求数量, $\eta_u(\theta, \tau)$ 表示在时刻 τ 应给具有优先级 θ 的设计活动分配的第 u 类资源的数量。更新 a_{hi} 的工作量为 $c_{hi}(\tau) = c_{hi}(\tau - 1) + \mu(\theta, \tau)$, 更新 a_{hi} 的时差为 $s_{hi}(\tau) = s_{hi}(\tau - 1) - (1 - \mu(\theta, \tau))$。

规则 8 在时刻 τ, 对于相应令牌(及有色令牌) 在库所 p_6 中的设计活动 a_{hi}(设其优先级为 θ), 计算 $\mu(\theta, \tau) = \min\{\eta_u(\theta, \tau)/r_{\theta u}, u = 1, \cdots, q\}$, 更新 a_{hi} 的修改工作量为 $c'_{hi}(\tau) = c'_{hi}(\tau - 1) + \mu(\theta, \tau)$, 更新 a_{hi} 的时差为 $s_{hi}(\tau) = s_{hi}(\tau - 1) - 1$。

4) 仿真终止规则:

规则 9 若所有设计活动的正常设计活动和修改设计活动都已完成, 即所有相关的令牌都在库所 p_7 中, 则仿真完成。

3.2 仿真流程

仿真的输入为初始产品开发计划及有关数据。其中, 初始产品开发计划的制订问题可建模为一个有资源约束的项目调度问题, 解之即可求得初始计划; 各种相关数据可通过统计或预测的方法获得。仿

真的输出为每个设计活动的开始和结束时间及每类资源的利用率。仿真流程如下:

步骤 0: 初始化。令 τ 表示当前时间, 并令普通库所内的初始标识为

$$M_0(p_0) = \{a_{hi}, h = 1, \cdots, g, i = 2, \cdots, s_h\}$$

$$M_0(p_1) = \{a_{h1}, h = 1, \cdots, g\}$$

$$M_0(p_2) = \{m_{uv}, u = 1, \cdots, q, v = 1, \cdots, r_u\}$$

$$M_0(p_3) = M_0(p_4) = M_0(p_5) =$$

$$M_0(p_6) = M_0(p_7) = M_0(p_8) = \varnothing$$

步骤 1: 确定可施行的设计活动。

步骤 2: 根据规则 1～ 规则 3, 对可施行的设计活动(即相应令牌在库所 p_1, p_3, p_5 或 p_6 的活动) 排序。

步骤 3: 按优先级从高到低的顺序, 确定应分配给可施行的设计活动的各类资源数;

1) 令当前时间为 τ, 根据规则 4, 判断是否需要重新分配资源, 若需要, 则令 $\theta = 1$ 并转 2), 若不需要则转步骤 5;

2) 根据规则 5, 确定应分配给可施行的设计活动的各类资源数;

3) 若所有可施行的设计活动应分配的各类资源数都已确定, 则转步骤4, 否则令$\theta \leftarrow \theta + 1$并转2)。

步骤 4: 为可施行的设计活动分配资源;

1) 令 $\theta = 1$;

2) 对具有优先级 θ 的设计活动 a_{hi}, 令 $\xi_{hiu} = 0$, $u = 1, \cdots, q$, 取 $u = 1$, 这里 ξ_{hiu} 表示已分配给 a_{hi} 的第 u 类资源的数量;

3) 根据规则 6, 为可施行的设计活动分配资源, 若对所有的 $u = 1, \cdots, q$, $\xi_{hiu} = \eta_{hiu}$ 成立, 则转 4), 否则令 $u \leftarrow u + 1$ 并重复 3), 这里 η_{hiu} 表示应分配给 a_{hi} 的第 u 类资源的数量;

4) 若所有可施行的设计活动都已分配所需要的资源, 则转步骤 5, 否则令 $\theta \leftarrow \theta + 1$ 并转 2)。

步骤 5: 根据第 3 类规则改变各设计活动和资源的状态。

步骤 6: 根据规则 9, 判断产品开发过程的仿真是否结束, 若结束则转步骤 7, 若未结束则令 $\tau \leftarrow \tau + 1$ 并转步骤 2。

步骤 7: 仿真结束。

4 仿真算例

现以并行工程环境中某汽车传动系统的开发过程为例, 具体说明基于 ESHL EP-N 的建模、调度与仿真方法。为简单起见, 假定只有一个产品开发项目在进行。

汽车传动系统的开发过程可分为 3 个阶段: 总体方案设计、分系统方案设计和详细设计。传动系统由 4 个分系统组成: 传动轴分系统、速度调节分系统、方向调节分系统和制动分系统。每个阶段包含多个设计活动。在并行工程环境中, 产品设计活动及其相关的工艺设计活动并行进行, 通过小范围的设计修改迭代相互影响。因此, 可用"产品 - 工艺设计活动对"对其描述, 并用产品 - 工艺设计活动对网络来描述传动系统的整个开发过程(见图 2)。图 2 中, 活动对 1 表示总体方案设计, 活动对 2～ 5 表示 4 个分系统方案设计, 活动对 6～ 9 表示 4 个分系统的详细设计, 活动 10 为虚活动, 表示产品开发项目结束。

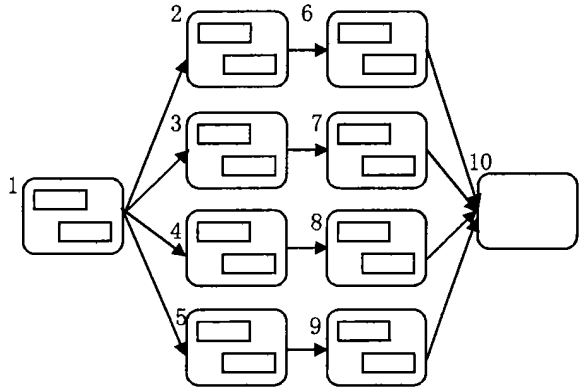

图 2　产品 - 工艺设计活动对网络

在每个产品 - 工艺设计活动对中, 产品设计活动和工艺设计活动都可分解为一系列子活动。当产品设计活动的某子活动结束时, 其设计结果传递给工艺设计活动; 当工艺设计活动的某子活动结束时, 产品设计活动的错误有可能被发现, 于是相关的产品和工艺设计活动则需要修改。

本例所需的仿真输入数据包括: 1) 各设计活动(正常和修改设计) 时间服从 β- 分布的参数; 2) 各子活动工作量占整个设计活动工作量的比例; 3) 可用的各类资源数量; 4) 各设计活动需要的各类资源数量; 5) 各资源的缺席结束(开始) 时间与下一次缺席开始(结束) 时间的间隔服从负指数分布的参数; 6) 每个工艺设计子活动结束时设计修改迭代发生的概率; 7) 初始产品开发计划。

初始产品开发计划如表 1 第 3 列和第 4 列所示。初始计划中产品开发项目的完成时间均值为 158. 0 天, 资源总平均利用率为 80. 4%。只要用第 i 个活动

对中第 j 个设计活动的第 k 个子活动代替第 h 个项目中第 i 个设计活动, 便可将上述仿真流程和调度规则应用于本例。仿真程序用 C 语言编写, 并在 Pentium 133MHz 计算机(内存 32M) 上运行。

用上述初始计划和参数作为仿真程序的输入, 进行第一轮 15 次仿真, 所用时间为 109.0s。仿真结果如表 1 第 5 列和第 6 列所示。产品开发项目完成时间均值为 184.3 天, 仿真结果与初始计划项目完成时间均值的相对误差为(184.3 - 158.0)/158.0 = 16.6%。4 类资源的平均利用率分别为 66.4%, 73.8%, 75.0% 和 74.8%, 资源总平均利用率为 70.2%。从表 1 可见, 仿真结果中各设计活动的平均开始与结束时间迟于初始计划, 资源总平均利用率低于初始计划。这是由于在制订初始产品开发计划时, 为简化问题而忽略了资源的离散性及资源缺席的情况。因此需要根据仿真结果对初始计划进行修正。

事实上, 初始计划仿真结果中的设计活动开始和结束时间均值可直接作为实际产品开发计划, 用以指导和协调设计活动。把初始计划仿真结果及有关参数作为仿真程序的输入, 进行第二轮 15 次仿真, 仿真时间仍为 109.0s, 结果如表 1 第 7 列和第 8 列所示。从第二轮仿真结果可见, 产品开发项目的平均完成时间为 188.1 天, 4 类资源的平均利用率分别为 66.6%, 72.3%, 73.0% 和 73.4%, 资源的总平均利用率为 69.5%。可见第二轮仿真结果中项目完成时间及资源利用率与实际计划(即第一轮仿真结果)很接近。因此, 实际的产品开发计划是可行的, 调度规则是有效的。

表 1 开发项目的初始计划、实际计划和实际计划仿真(时间单位: 天)

活动对号	活动号	初始计划		实际计划(初始计划仿真)		实际计划仿真	
		开始时间均值	结束时间均值	开始时间均值	结束时间均值	开始时间均值	结束时间均值
1	1	0.0	47.9	0.0	41.9	0.0	47.0
1	2	0.0	47.9	5.0	47.3	5.0	52.4
2	1	47.9	106.8	47.4	113.9	52.5	126.0
2	2	47.9	106.8	78.8	120.6	82.4	129.5
3	1	47.9	82.8	47.4	84.1	52.5	89.5
3	2	47.9	82.8	52.3	88.3	57.4	93.4
4	1	47.9	85.6	47.4	87.9	52.5	91.7
4	2	47.9	85.6	52.4	92.0	57.3	95.3
5	1	47.9	83.6	47.4	88.1	52.5	91.1
5	2	47.9	83.6	53.4	90.1	59.2	95.0
6	1	106.8	156.0	120.7	174.2	129.6	182.8
6	2	106.8	156.0	124.4	184.3	133.5	188.1
7	1	82.8	131.6	88.4	131.0	93.5	128.7
7	2	82.8	131.6	94.3	134.7	98.9	132.0
8	1	85.6	155.0	92.1	166.3	95.4	156.9
8	2	85.6	155.0	105.1	174.8	110.2	164.8
9	1	83.6	158.0	90.2	175.2	95.1	173.5
9	2	83.6	158.0	114.7	180.5	121.2	178.0

5 结 语

本文用 ESHL EP-N 建立了产品开发过程的模型, 并在此基础上给出了产品开发过程的调度仿真方法。ESHL EP-N 模型可精确地描述产品开发过程的行为, 特别是设计迭代和资源分配的情况。由于其简洁性和具有较强的描述与决策能力, 这一模型很便于计算机实现。在此模型基础上, 以初始产品开发计划为输入, 根据本文给出的 4 种调度规则(即活动排序、资源分配、状态改变和仿真终止规则)及仿真流程, 可对产品开发过程进行仿真, 并根据仿真结果获得实际可行的产品开发计划。算例的仿真结果表明, 这 4 种调度规则对实现产品开发计划是有效的, 通过仿真获得的实际产品开发计划是可行的。

(下转第 295 页)

待车流队长很长, 即已发生交通拥塞。

从仿真结果可以看出, 多智能体方法的控制效果优于其它两种方法。当网络交通流并不拥挤时, 其优势并不明显; 但当网络流量较大时, 本文方法在其它两种方法产生拥塞时仍能有效地控制交通流量。M 3 只考虑了单路口情况, 显然不能适应交通流的动态变化, 只适用于网络车流量稀疏的情况。M 2 考虑了其它交叉口的优化, 对一般网络车流量情况下能较好地进行网络优化控制, 但实时性较差, 不能适应拥塞时的网络流量优化和处理突发事件。

6 结　　论

本文在分布式多智能体环境下对城市交通网络控制的建模和协调进行了深入研究, 应用递归建模方法进行网络中消除交通拥塞的高层次决策, 最大程度地减少或消除了交通网络控制中对各节点之间大量通讯的需求。提出采用贝叶斯学习方法与 RMM 框架结构相结合来跟踪交通网络中的动态变化, 各智能体之间实时更新信息, 增强了决策的可靠性。基于 RMM 和贝叶斯学习建立了简单的城市交通多智能体控制系统, 仿真研究取得了较好的效果, 对 ITS 的发展具有重要意义。

参考文献:

[1] 冯蔚东, 贺国光, 刘豹. 交通流理论评述[J]. 系统工程学报, 1998, 3(1): 71-87.

[2] 王亦兵, 韩曾晋, 贺国光. 城市高速公路交通控制综述[J]. 自动化学报, 1998, 7(1): 485-496.

[3] B Burmeister, A Haddadi, G Matylis. Applications of multi-agent systems in traffic and transportation [J]. IEE Trans on Software Engineering, 1997, 144(1): 51-60.

[4] C V Goldman, J S Rosenschein. Mutual supervised learning in multi-agent systems [A]. Distributed AI [M]. 1996. 85-96.

[5] N V Findler. Distributed control of collaborating and learning expert systems for street traffic signals [A]. IFAC Distributed Intelligence Systems [C]. Pergamon Press, 1991. 125-130.

[6] Gerhard Weib. Introduction to distributed artificial intelligence[M]. Cambridge: MIT Press, 1998.

[7] P J Gmytrasiewicz, E H Durfee. A rigorous, operational formalization of recursive modeling[A]. ICMAS-95[C]. 1995. 125-132.

[8] Martin Anthony, Norman Biggs. Computational learning theory [M]. Cambridge: Cambridge University Press, 1992.

(上接第 286 页)

参考文献:

[1] Smith R P, Eppinger S D. Identifying controlling features of engineering design iteration [J]. Management Science, 1997, 43(3): 276-293.

[2] Kusiak A, Larson T N, Wang J. Reengineering of design and manufacturing process[J]. Computers and Industrial Engineering, 1994, 26(3): 521-536.

[3] Luh P B, Liu F, Moser B. Scheduling of design projects with uncertainty number of iterations[J]. European J of Operational Research, 1999, 113(3): 575-592.

[4] Zhou M C, Di Cesare F, Desrochers A A. A hybrid methodology for synthesis of Petri net models for manufacturing systems [J]. IEEE Trans on Robotics and Automation, 1992, 8(3): 350-361.

[5] Yan H S, Wang N S, Cui X Y *et al*. Modeling, scheduling and control of flexible manufacturing systems by extended high-level evaluation Petri nets[J]. IIE Trans, 1997, 29(2): 147-158.

[6] Yan H S, Wang N S, Zhang J G *et al*. Modelling, scheduling and simulation of flexible manufacturing systems using extended stochastic high-level evaluation Petri nets[J]. Robotics and Computer-integrated Manufacturing, 1998, 14(2): 121-140.

[7] Yan H S, Jiang J. Agile concurrent engineering[J]. Integrated Manufacturing Systems, 1999, 10(2): 103-112.