第十届湖南省研究生数学建模竞赛承诺书

我们仔细阅读了湖南省高校研究生数学建模竞赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们完全清楚，在竞赛中必须合法合规地使用文献资料、软件工具和 AI 工具，不能有任何侵犯知识产权的行为。否则我们将失去评奖资格，并可能受到严肃处理。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们授权湖南省研究生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

我们参赛选择的题号是（从组委会提供的赛题中选择一项填写）：A 题

我们的参赛编号（请填写完整参赛编号）：A202518001012

所属学校（请填写完整的全名）：国防科技大学

参赛队员（打印后签名）：1. 李典

2. 叶超然

3. 石钧方

指导教师或指导教师组负责人（打印后签名）：高勋章

日期：2025 年 8 月 27 日

（请勿改动此页内容和格式。以上内容请仔细核对，如填写错误，论文可能被取消评奖资格。）

基于 NSGA-II 的测试资源调度优化问题

摘要

随着现代工业系统复杂性提升，大型装置在能源、制造等关键领域应用愈发广泛，其测试环节面临测试周期长、漏判/误判概率高、资源利用率低等问题，传统调度方式已难以满足需求。本文针对带随机扰动的多目标并行机混合流水作业测试调度问题，综合考虑设备故障、子系统问题、测手差错等随机事件及设备更换、班次时间、重测机制等约束，构建基于 NSGA-II 算法与蒙特卡洛模拟的多维度数学模型求解。

针对问题一:针对综合测试问题指向性分析与概率计算问题，基于贝叶斯推断与全概率公式，结合子系统先验问题概率、测手漏判/误判行为，推导综合测试问题指向各子系统（A/B/C/D）的比例参数 λ_1~λ_4（分别为 0.224、0.357、0.120、0.300），并计算出综合测试发现问题概率为 3.30×10^{-3}，为后续测试可靠性评估提供基础参数。

针对问题二:针对单个测试分队测试 100 个装置的计划安排问题，以最小化任务完成平均天数和总漏判概率为目标，构建基于 NSGA-II 算法的多目标优化模型。采用“离散＋连续”的混合编码表征测试台分配、测试顺序、设备更换时间等决策变量，结合蒙特卡洛模拟消除随机扰动。求解得到 Pareto 最优解集，其中均衡解的任务完成平均天数为 51.86 天，通过测试装置数 99 个，总漏判概率 0.0012，总误判概率 0.0547，各测试组有效工作时间比均超 0.32，资源利用率均衡。

针对问题二:针对双分队接续倒班测试问题，在问题二的模型基础上，采用“枚举 K 值+NSGA-II 优化+蒙特卡洛模拟”策略。在 $9\le K\le12$ 小时的班次时间范围内，以 0.5 小时步长，遍历 7 个候选 K 值，对每个 K 值生成最优测试计划并统计指标。结果显示 K=12.0 小时时性能最优：任务完成天数 31.35 天，较 K=9.0 小时缩短 36%。通过装置数 99.2 个，总漏判概率稳定在 0.0012，各测试组有效工作时间比随 K 值增大呈上升趋势，E 组最高达 6.86%。

针对问题二:针对测试任务影响因素分析与改进建议问题，基于问题三模型拆解设备故障、班次时间、流程约束等因素对完成时间的作用路径，通过灵敏度分析量化影响权重。结果表明，设备故障通过中断重测延长工序时间，班次时间 K 在高灵敏度区对完成时间影响显著。据此提出重点关注 E/A 工序设备的设备维护优化、K 值设定 10.5 小时、提前停止启动新工序的班末工序管理等改进措施，提升测试效率与可靠性。

最后，通过离散事件仿真模型与本文所建模型的结果对比、稳定性分析等手段，对模型进行了系统且深入的分析与评价。结果表明，将 NSGA-II 算法与蒙特卡洛模拟相结合，能够有效应对测试过程中的随机扰动与多重约束，所生成的测试调度方案可为复杂工业系统测试决策提供科学支撑。同时，该模型在多设备协同、多工序衔接、多随机事件干扰的生产调度场景中具有良好的通用性，可进一步推广应用于类似调度问题。

关键词：多目标优化；贝叶斯推断；NSGA-II 算法；蒙特卡洛模拟；离散事件仿真

目录

1 问题综述

1.1 问题背景

随着现代工业系统复杂性的不断提升，大型装置在能源、制造、航空航天等关键领域的应用日益广泛，其运行可靠性直接关系到生产安全与经济效益。这类装置通常由多个功能关联的子系统构成，任何一个子系统的失效都可能引发连锁故障，因此对测试环节的可靠性与效率提出了极高要求[1]。在装置需求量持续增长、专业测试资源供给有限的背景下，传统的"单线程顺序测试+人工调度"方式容易产生测试周期过长、漏判/误判概率高、资源利用率低等问题，严重影响了装置交付效率与质量控制水平。

目前，已有众多学者针对复杂系统测试调度问题开展了深入研究，提出了多种优化算法来提升测试效率与可靠性。传统的测试调度算法包括基于贪心策略的优先级调度算法、动态规划算法、遗传算法等，近几年随着高性能计算设备的普及，还涌现出强化学习算法、神经网络预测算法等智能优化方法。这些算法虽然能够快速生成初步的测试方案，但其核心调度框架多基于"理想无干扰"假设——仅保证大多数情况下的时间分配合理性，却难以应对实际测试中频繁出现的设备故障、子系统误判/漏判、异常重测等复杂动态场景。例如，当测试设备发生故障时，传统算法无法自动触发设备更换逻辑；当测手出现误判时，现有模型难以量化其对整体漏判概率的影响。这些局限性导致实际测试过程中常出现计划外中断、资源冲突、指标偏离等问题，亟需建立更贴合实际的数学模型，综合考虑设备状态、测试异常、人员操作等多维度因素，制定兼顾效率与可靠性的动态测试方案。测试示意图 1 测试场景示意图如图 1 所示

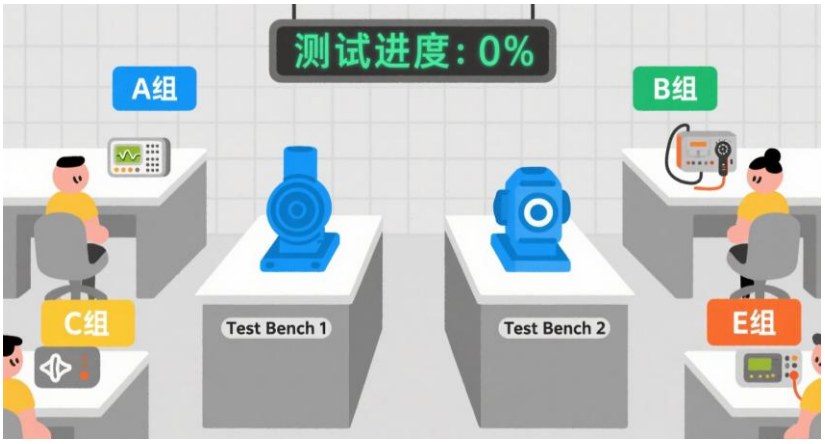


图 1 测试场景示意图

1.2 问题重述

本题是一个带随机扰动的多目标并行机混合流水作业调度问题。该系统包含两个并行测试台、四个专业测试小组（A/B/C/E），每个装置需依次通过 A、B、C 子系统测试和综合测试 E。测试过程中可能发生四类随机事件（Y1 设备故障、Y2 子系统问题、Y3 测手误判/漏判、Y4 综合测试问题），且需考虑设备更换、班次时间、重测机制等复杂约束。优化目标包括最小化任务完成天数和总漏判概率，决策变量既包括离散的测试顺序与设备更换策略，也包括连续的设备更换时间，属于典型的混合变量多目标优化问题。

已知条件如下：

1.测试大厅有两个测试台，可同时测试 2 个装置；

2.4 个测试小组（A/B/C/E）独立工作，装置需依次通过 A、B、C 测试后方可进入 E 测试；

3.任一测试未通过需重测，连续两次失败则退出；

4.测试过程中可能发生四类随机事件：设备故障（Y1）、子系统问题（Y2）、测手差错（Y31 误判/Y32 漏判）、综合测试问题（Y4）；

5.中断测试需重新开始；

6.设备故障概率随使用时间分段增长，需在 120-240 小时之间更换；

7.各子系统问题概率、测手差错概率等已知。

需要从测试效率、资源利用率、质量控制和工作负荷角度考虑, 解决以下 4 个问题::

(1) **问题 1**: 针对综合测试问题指向性分析与概率计算问题，需建立数学模型，推导综合测试发现问题指向各子系统（A/B/C/D）的比例参数 λ1~λ4（λ1+λ2+λ3+λ4=1），并构建综合测试测出系统存在问题的概率计算公式。首先明确 λ_i 表示在综合测试检出问题时，问题来源于子系统 i 的后验概率。结合子系统先验问题概率、测手漏判/误判行为，以及综合测试的检测特性，通过全概率公式与贝叶斯推断建立 λ_i 与综合测试出现问题概率的表达式，代入给定参数计算具体数值。

(2) **问题 2**: 针对单个测试分队测试 100 个大型装置的计划安排问题，在每日一个不超过 12 小时班次、测试中断需重新开始的约束下，制定测试工作计划，计算任务完成平均天数(T)、通过测试装置的平均数目(S)、总漏判概率(PL)、总误判概率(PW)以及各测试组有效工作时间比(YXB1-YXB4)等统计指标。

(3) **问题 3**: 针对两个测试分队接续倒班测试第二批 100 个装置的任务安排问题，在每个班次工作 K 小时（9≤K≤12，以半小时为单位）且同工序两个班次共用一套测试装备的条件下，确定最优 K 值，制定测试工作计划，并计算任务完成平均天数、通过测试装置的平均数目、总漏判概率、总误判概率及各测试组有效工作时间比(YXB)等统计指标。

(4) **问题 4**: 针对问题三中测试任务安排的各种影响因素，分析各因素对测试任务平均完成时间的影响规律，并据此向主管部门提出测试工作的改进建议。

2 模型假设与符号说明

2.1 模型基本假设

(1) 假设重测指的是对同一装备的 ABC 中的错误系统进行重测，而非重新开始测试所有系统。

(2) 假设调试时间不算入到使用时间中。

(3) 假设综合测试测出各子系统存在问题的能力相同且存在问题一定能检出。

(4) 假设不存在多个系统同时出现问题的情况

(5) 假设 A,B,C 三个子系统相互独立

(6) 假设测试台空闲后，可立即运入下一个待测试装置（假设第二批装置已全部到位，无供应延迟）；运入 / 运出过程不占用工位时间仅占用测试台 “放置时间”。

(7) 假设装置退出测试（连续 2 次未通过）后，立即运出，空出的测试台和工位资源优先分配给未测试的装置

(8) 假设设备累计使用时间 = 其实际用于测试的时间总和（不包括调试、更换、空闲时间）；跨班次使用时，时间连续累计

(9) 假设运输资源充足：装置运入 / 运出可随时进行，无运输人员或设备等待时间，运输时间严格为 0.5h / 次

(10) 假设 Y1（设备故障）、Y2（子系统问题）、Y3（测手差错）、Y4（综合测试问题）相互独立；不同装置、不同工序的随机事件也独立

2.2 符号说明

本文定义了如下【数字】个使用次数较多的符号，其余符号在使用时注明.

表1 符号说明

符号	含义	单位
$P_{defect0,i}$	子系统 i 本身存在问题的先验概率(i=A,B,C,D)	无
$P_{error,i}$	测手差错总概率(i=A,B,C,E)	无
$P_{false,i}$	误判概率(i=A,B,C,E)	无
$P_{miss,i}$	漏判概率(i=A,B,C,E)	无
$P_{defect,i}$	子系统实际存在问题的概率(i=A,B,C,D)	无
P_{total}	综合测试发现问题概率	无
λ_i	指向各子系统(A/B/C/D)的比例系数(i=1,2,3,4)	无
x_{split}	测试台 1 分配的装置数量	个
ID_1	测试台 1 装置序列	无
ID_2	测试台 2 装置序列	无
S_A	A 组测试顺序	无
S_B	B 组测试顺序	无
S_C	C 组测试顺序	无
S_D	E 组测试顺序	无
$t_{replace,A}$	A 设备更换时间	小时(h)
$t_{replace,B}$	B 设备更换时间	小时(h)
$t_{replace,C}$	C 设备更换时间	小时(h)
$t_{replace,D}$	E 设备更换时间	小时(h)
$F_{fail,i}$	累计故障概率(i=A,B,C,E)	无
$F_i(t)$	故障概率函数(i=A,B,C,E)	无
f_1	任务完成平均天数函数	
$T^{(k)}$	表示第 k 次模拟的总任务时间	小时（h）
f_2	总漏判概率	无
$t_{test.i}^{(k)}$	第 k 次模拟中第 i 个装置的测试总时间	h
$t_{in.i}^{(k)}$	表示第 k 次模拟中第 i 个装置的运入时间	h
$t_{out.i}^{(k)}$	表示第 k 次模拟中第 i 个装置的运出时间	h
$t_{shift.i}^{(k)}$	表示第 k 次模拟中第 i 个装置因跨班次产生的等待时间	h
$N_{pass}^{(k)}$	表示第 k 次模拟中通过测试的装置总数	个

3 问题分析

3.1 问题一分析

本问聚焦于综合测试环节中检测问题的指向性分配与比例参数计算问题。根据题目描述，综合测试（E）用于检测已通过 A、B、C 子系统测试的装置是否存在联接问题（子

系统 D）或其他潜在缺陷。其核心在于：当综合测试报告“系统有问题”时，需明确该问题具体来源于哪个子系统（A/B/C/D），即确定比例参数 λ_1~λ_4（满足 $\lambda_1+\lambda_2+\lambda_3+\lambda_4=1$），并构建综合测试测出系统存在问题的总概率计算公式。

对题干分析可知，综合测试报出“有问题”的情形可分为两类：一是系统确实存在故障且被正确检出；二是系统正常但被误判为有故障（误判率 50%）。考虑第一种情况，在综合测试环节中，λ_i 表示当检测出系统问题时，问题实际来源于子系统 i（A、B、C 或 D）的后验概率，即 λ_i= P(问题来自子系统 i| 综合测试报有问题)。P_{total}代表综合测试能够准确识别出系统中存在实际问题且未被漏判的概率。在建模过程中，应首先计算各子系统实际存在问题的概率$P_{defect,i}$，综合考虑测试过程中的漏判与重测机制，随后借助全概率公式和贝叶斯方法推导出 λ_i 和 P_{total} 的表达式，最终代入给定的子系统故障率和测手差错率等参数，计算出具体的 λ_i 和 P_{total} 值。

根据题目中关于子系统问题概率、测手差错概率、综合测试指向性等描述，我们做出如下合理假设：

假设 1：子系统 A、B、C、D 的问题发生相互独立（基于题目中“子系统 D 问题概率为 0.1%”等独立描述）；

假设 2：综合测试对各子系统问题的检出能力相同，且存在问题一定能被检出（题目中未明确区分检出能力差异）；

假设 3：测手差错行为独立于子系统问题，且误判与漏判各占差错的一半（题目中“测手差错”未进一步细分，故按 50%划分）；

假设 4：不存在多个子系统同时出现问题的情况（因其概率极小，可忽略）。

3.2 问题二分析

本问聚焦为 100 个大型装置制定测试计划，需考虑两个工作台和 4 个测试小组的工作调配，同时面临多种约束条件，目标是使任务完成平均天数最小化以及总漏判概率最小化，这是一个典型的多目标优化问题。根据题设条件，需在测试时间（正常/调试/运输）、设备工作约束（120-240h 更换规则）、随机事件（设备故障/子系统问题/测手误判）及流程要求（E 测试必须在 A/B/C 完成后、中断重试等）等复杂约束下，明确测试台分配、小组测试顺序、设备更换时间等决策变量的可行域。可以通过 NSGA-II 算法构建包含这些决策变量的染色体编码，对种群进行迭代优化：先初始化随机方案，经非支配排序筛选优势个体，计算拥挤距离保持多样性，再通过选择/交叉/变异生成新种群，最终得到 Pareto 最优解集；求解时需通过蒙特卡洛模拟多次试验统计目标均值。最终可从解集中根据实际需求（如侧重时间或漏判率）筛选最优测试计划，并由此计算题目要求的各项指标。

根据问题 2 的题设条件，进行如下的假设：

假设 1.测试台空闲后，可立即运入下一个待测试装置（假设第二批装置已全部到位，无供应延迟）；运入/运出过程不占用工位时间（仅占用测试台 “放置时间”）

假设 2.装置退出测试（连续 2 次未通过）后，立即运出，空出的测试台和工位资源优先分配给未测试的装置

假设 3.设备累计使用时间 = 其实际用于测试的时间总和（不包括调试、更换、空闲时间）；跨班次使用时，时间连续累计

假设 4.运输资源充足：装置运入 / 运出可随时进行，无运输人员或设备等待时间，运输时间严格为 0.5h / 次

假设 5.Y1（设备故障）、Y2（子系统问题）、Y3（测手差错）、Y4（综合测试问题）相互独立；不同装置、不同工序的随机事件也独立

假设 6.若某工序测试的 “启动时间 + 预计测试时间”> 当前班次结束时间，视为“班次中断”，该工序需在下一班次重新开始（如 K=9h 时，A 测试从班次第 8h 启动，需 2.5h，超过班次结束时间，中断后下一班次重新测 2.5h）

3.3 问题三分析

本问聚焦于双分队倒班测试场景下的最优班次工作时间（K 值）决策与测试计划制定，核心目标是在 9-12 小时的班次时间范围内（步长 0.5 小时），通过多目标优化平衡任务完成时间、总漏判概率及资源利用效率，最终确定最优 K 值并计算相关统计指标（通过装置数、有效工作时间比等）。相对于问题二的单班次测试场景，其在 24 小时接续，K 值可变，跨班次积累设备工作事件需要更精准的更换时机。本部分的解决问题框架为采用“枚举+优化”两阶段方法：首先，枚举 K 值：遍历 7 个候选 K 值，针对每个 K 值执行 NSGA-II 多目标优化算法生成测试计划；其次，蒙特卡洛模拟：对每个计划进行 1000 次模拟，统计任务时间、漏判概率等指标；最后 Pareto 最优选择：综合权衡时间与可靠性，推荐最优 K 值。

根据问题 2 的题设条件，进行如下的假设：

假设 1：设备工作时间累积规则：设备累计使用时间=实际测试时间总和（不含调试、更换、空闲时间），跨班次连续累计；

假设 2：跨班次中断处理：若工序启动时间+预计测试时间>当前班次结束时间，该工序需在下一班次重新开始（如 K=9h 时，8h 启动的 2.5h 测试需中断后重测）；

假设 3：运输与交接假设：运输时间固定为 0.5h/次，资源充足且不占用工位时间；装置退出测试后立即运出，空出资源优先分配给未测试装置；

假设 4：随机事件独立性：设备故障（Y1）、子系统问题（Y2）、测手差错（Y3）等随机事件相互独立，不同装置/工序的事件亦独立；

假设 5：分队效率一致性：两个分队的测试能力、差错率无差异，仅工作时段不同。

3.4 问题四分析

本问聚焦于问题 3 双分队倒班测试场景下，各因素对测试任务平均完成时间的影响机制分析，并据此提出测试工作改进建议。根据题目描述，问题 3 已明确“两个分队接续倒班、同工序共用测试装备、K 值为 9-12 小时且以半小时为单位”的测试模式，问题 4 需在此基础上，系统拆解设备状态、流程规则、倒班调度等关键因素与平均完成时间的关联，量化影响程度并形成优化方案。

对题干及问题 3 背景分析可知，测试任务平均完成时间的核心影响逻辑为“因素→测试流程耗时变量→总时间”：设备故障（Y1）通过中断重测、强制更换及调试耗时延长单工序时间；倒班安排通过 K 值设定影响倒班次数与班末中断概率，进而增加额外重测耗时；测试流程的并行（A/B/C）与串行（E 工序）约束则决定单装置基础流程时长。在分析过程中，应首先基于问题 3 的时间计算模型，拆解各因素（如设备累积故障概率、K 值、调试时间等）的作用路径，再通过灵敏度分析量化不同因素的影响权重，最终结合测试任务“效率优先、兼顾可靠性”的目标，提出针对性改进建议。

根据题目中关于设备故障、倒班规则、测试流程等描述，我们做出如下合理假设：

假设 1：设备故障严格遵循“120h/240h”累积概率划分，故障发生后立即更换且调试时间固定（基于文档 Y1 的明确规定）；

假设 2：倒班交接仅导致未完成工序重新开始，已完成工序不受影响（题目中“中断不得接续需重新开始”的规则延伸）；

假设 3：运输过程与测试完全并行，不产生等待耗时（文档中“运输不影响大厅内测试”的明确说明）；

假设 4：各工序测试、设备更换、倒班交接等环节互不干扰，无资源冲突（题目未提及资源竞争，故简化处理）。

4 模型建立与求解

4.1 问题一的模型建立及求解

4.1.1 问题一的模型建立

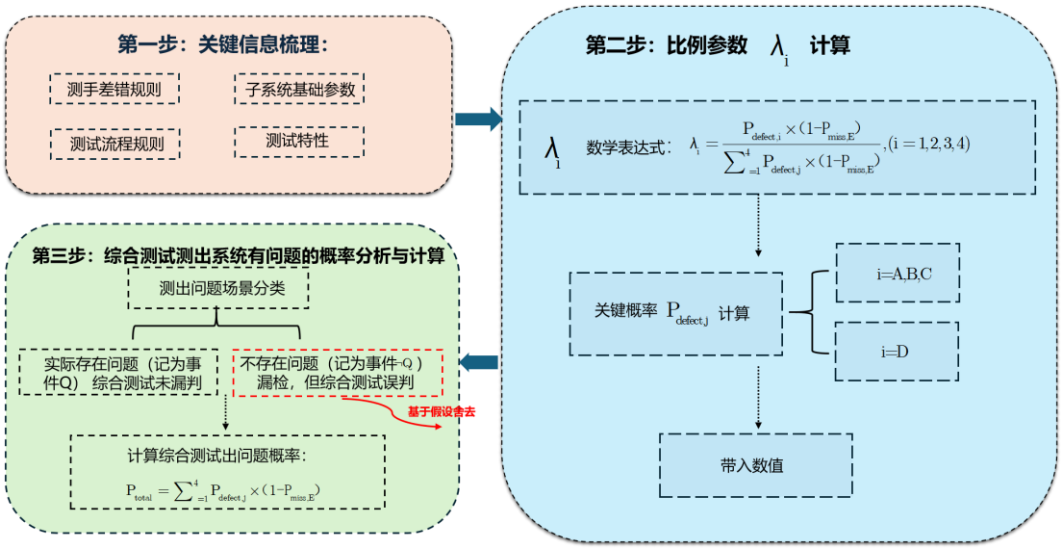


图 2 问题一求解流程图

根据第 3 节对问题的分析，本题聚焦综合测试"系统有问题"结果下各子系统有问题的后验概率的计算。

根据题设条件，得到以下概率模型：

1. 子系统本身存在问题的先验概率：

$$P_{defect0,i}=\begin{cases}0.025, & i=A\\ 0.03, & i=B\\ 0.02, & i=C\\ 0.001, & i=D\end{cases} \tag{1}$$

2. 测手差错概率：

测手差错总概率：

$$P_{error,i}=\begin{cases}0.03, & i=A\\ 0.04, & i=B\\ 0.02, & i=C\\ 0.02, & i=E\end{cases} \tag{2}$$

误判概率（系统无问题但判为有问题）：

$$P_{false,i} = 0.5 \times P_{error,i} = \begin{cases} 0.15, & i = A \\ 0.02, & i = B \\ 0.01, & i = C \\ 0.01, & i = E \end{cases} \tag{3}$$

漏判概率（系统有问题但判为无问题）：

$$P_{miss,i} = 0.5 \times P_{error,i} = \begin{cases} 0.15, & i = A \\ 0.02, & i = B \\ 0.01, & i = C \\ 0.01, & i = E \end{cases} \tag{4}$$

3. 子系统实际存在问题的概率

子系统实际存在问题说明在 A、B、C 子系统的检测阶段存在测手漏检的情况，分为第一次直接漏检和第一次正确检出但第二次漏检两种情况，因此得到如下的概率表达式：

$$P_{defect,i} = \begin{cases} P_{defect0,i} \times (P_{miss,i} + P_{miss,i} \times (1 - P_{miss,i})), & i = A, B, C \\ 0.01, & i = D \end{cases} \tag{5}$$

带入数值计算上述式子，可以得到如下的结果：

$$P_{defect,i} = \begin{cases} 7.44375 \times 10^{-4}, & i = A \\ 1.188 \times 10^{-3}, & i = B \\ 3.98 \times 10^{-4}, & i = C \\ 1 \times 10^{-3}, & i = D \end{cases} \tag{6}$$

由于 A,B,C,D 系统相互独立，故两个子系统同时存在问题的概率为：

$$P_{defect,i}P_{defect,j} \le P_{defect,B}P_{defect,D} = 1.188 \times 10^{-3} \times 1 \times 10^{-3} = 1.188 \times 10^{-6} \ll P_{defect,i} \tag{7}$$

所以可以假设不存在多个系统同时出现问题的情况。

4. 比例参数λ_i计算

由于各子系统问题在综合测试发现问题中的指向比例参数 λ_1（A）、λ_2（B）、λ_3（C）、λ_4（D）表示综合存在问题指向各系统的概率，且满足 $\lambda_1+\lambda_2+\lambda_3+\lambda_4=1$，故有：

$$\lambda_i = \frac{i \text{ 系统实际有问题的概率}}{\text{各系统实际有问题的总概率}} = \frac{P_{defect,i} \times (1 - P_{miss,E})}{\sum_{j=1}^{4} P_{defect,j} \times (1 - P_{miss,E})}, \quad i = 1,2,3,4 \tag{8}$$

5. 综合测试发现问题概率

$$P_{total} = \sum_{i=1}^{4} P_{defect,i} \times (1 - P_{miss,E}) \tag{9}$$

4.1.2 问题一的求解

由式 6，式 8 可以计算出指向各子系统(A/B/C/D)的比例系数(保留三位小数）：

$$\lambda_i = \begin{cases} 0.224, & i = 1 \\ 0.357, & i = 2 \\ 0.120, & i = 3 \\ 0.300, & i = 4 \end{cases} \tag{10}$$

由式 4，式 6，式 9 可以计算出综合测试发现问题概率：

$$P_{total} = 3.30 \times 10^{-3} \tag{11}$$

表2 问题一求解结果

综合测试测出有问题时各部分所占比例				P_{total}
子系统 A	子系统 B	子系统 C	子系统 D	3.30×10^{-3}
0.224	0.357	0.120	0.300	

4.1.3 问题一结果分析

通过对问题一综合测试发现问题指向性比例参数 λ_1~λ_4及综合测试发现问题概率 P_{total}的计算，我们得到了以下重要结论：

1.指向性比例分布合理性分析：计算得到的比例参数 λ_1=0.224（指向子系统 A）、λ_2=0.357（指向子系统 B）、λ_3=0.120（指向子系统 C）、λ_4=0.300（指向子系统 D）符合各子系统先验故障率的分布规律。其中子系统 B 的比例最高（0.357），这与子系统 B 的最高先验故障率（3.0%）相吻合；子系统 C 的比例最低（0.120），也与其最低先验故障率（2.0%）相一致。这表明模型能够合理反映各子系统在实际测试中对综合测试结果的贡献程度。

2.漏判机制的影响：计算结果明显体现了测试过程中漏判行为对问题检测的削弱效应。各子系统实际存在问题概率$P_{defect,i}$均显著低于其先验故障率$P_{defect0,i}$，如子系统 A 的实际问题概率（7.44×10^{-4}）远低于其先验故障率（2.5%），这主要是由于测手漏判导致部分存在的问题未被及时发现而流入后续环节。

3.综合测试检出能力评估：综合测试发现问题概率P_{total}=3.30×10^{-3}表明，平均每 1000 次测试中约有 3.3 次能够正确检出系统存在的问题。这一数值相对较低，反映了在实际测试环境中，由于多层测试环节的漏判累积效应，真正存在问题并最终被综合测试检出的概率较小。

4.模型假设的验证：通过计算发现，多个子系统同时存在问题的概率极低（最大约为 1.188×10^{-6}），远小于单个子系统问题的概率，验证了"不存在多个系统同时出现问题"假设的合理性，说明该简化假设对模型精度影响可忽略不计。

5.实际应用意义：本研究建立的数学模型能够量化评估综合测试中各子系统的责任归属，为测试质量追溯和改进提供了理论依据。例如，针对指向性比例较高的子系统 B（λ_2=0.357），建议加强该子系统的测试资源配置和测手培训，以降低漏判率，提高整体测试效率

4.2 问题二的模型建立及求解

4.2.1 基于 NSGA-II 的测试任务规划模型

NSGA-II（非支配排序遗传算法 II）是一种用于解决多目标优化问题的算法。它是 NSGA 的改进版本，由 Deb 等人在 2002 年提出[2]。NSGA-II 的核心思想是通过非支配排序和拥挤距离来维持种群的多样性，同时采用精英策略来保留优秀个体，从而有效地搜索多目标问题的 Pareto 最优解集。NSGA-II 算法的主要特点包括：采用快速非支配排序方法，通过计算个体的支配数与被支配集合，将种群高效地划分为多个非支配前沿（时间复杂度为 $O(MN^2)$，其中 M 为目标数，N 为种群规模）；引入拥挤距离机制，评估同一非支配层中个体的分布密度，以维持种群的多样性；同时，运用精英保留策略，将父

代与子代合并排序，优先选择优质个体进入下一代，有效防止优秀解的丢失，从而提升算法的收敛性与解集的分布性。NSGA-II 算法广泛应用于各种多目标优化问题，如工程设计、资源分配、调度优化等领域。它能够有效地找到多个目标之间的权衡解，为决策者提供一系列可行的选择方案。

本题题设条件中的测试工作计划求解涉及两个工作台 4 个测试小组的工作调配，约束条件涉及测试顺序、设备更换时间、测试流程约束、随机事件触发约束等，优化目标有任务完成平均天数最小化，总漏判概率最小化等，是一个复杂条件下的多目标优化问题，可以选择基于 NSGA-II 算法建立任务规划模型求解。

4.2.1.1 设备故障概率模型

根据题目，设备故障遵循累计概率模型。累计概率是指在某个时间区间内至少发生一次故障的概率。

设备 i 在时间段[0,120h]内的**累计故障概率**：

$$F_{fail,i}^{(1)} = \begin{cases} 0.03, & i = A \\ 0.04, & i = B \\ 0.02, & i = C \\ 0.03, & i = E \end{cases} \tag{12}$$

设备 i 在时间段[120h,240h]内的**累计故障概率**：

$$F_{fail,i}^{(2)} = \begin{cases} 0.05, & i = A \\ 0.07, & i = B \\ 0.06, & i = C \\ 0.05, & i = E \end{cases} \tag{13}$$

假设故障遵循指数分布（常数故障率），则故障率参数为： - 第一阶段(0-120h)：

$$a_i^{(1)} = -\frac{\ln(1 - F_{fail,i}^{(1)})}{120} = \begin{cases} 2.54 \times 10^{-4}, & i = A \\ 3.40 \times 10^{-4}, & i = B \\ 1.68 \times 10^{-4}, & i = C \\ 2.54 \times 10^{-4}, & i = E \end{cases} \tag{14}$$

- 第二阶段(120h-240h)：

$$a_i^{(2)} = -\frac{\ln(1 - F_{fail,i}^{(2)})}{240} = \begin{cases} 2.14 \times 10^{-4}, & i = A \\ 3.03 \times 10^{-4}, & i = B \\ 2.58 \times 10^{-4}, & i = C \\ 2.14 \times 10^{-4}, & i = E \end{cases} \tag{15}$$

可靠性函数（在时间 t 内不发生故障的概率）：

$$R_i(t) = \begin{cases} \exp(-a_i^{(1)} \cdot t), & 0 \le t \le 120 \\ \exp(-a_i^{(2)} \cdot 120 - a_i^{(2)} \cdot (t - 120)), & 120 < t \le 240 \end{cases} \tag{16}$$

故障概率函数（在时间 t 内发生故障的概率，这是我们真正需要的）：

$$F_i(t) = 1 - R_i(t) = \begin{cases} 1 - \exp(-a_i^{(1)} \cdot t), & 0 \le t \le 120 \\ 1 - \exp(-a_i^{(2)} \cdot 120 - a_i^{(2)} \cdot (t - 120)), & 120 < t \le 240 \end{cases} \tag{17}$$

在代码实现中，我们先计算可靠性$R_i(t)$，然后用 1-$R_i(t)$得到故障概率$F_i(t)$，最后用故障概率$F_i(t)$来判断设备是否在测试期间发生故障。

4.2.1.2 解的编码

解的编码需覆盖第二问“测试计划制定”的核心决策变量，采用混合编码（离散 + 连续），将“测试台分配、测试顺序、设备更换时间”等抽象决策转化为可计算的“染色体向量”，编码结构与题设需求的对应关系如下表所示，根据上述编码结构，可以构建如下图所示的染色体，每条染色体代表测试计划的一个解。

表3 解编码结构表

编码段	编码段含义	变量符号	长度	取值范围
X	测试台 1 分配的装置数量	x_{split}	1	$x_{split} \in \{1,2,...,99\}$
ID1	测试台 1 装置序列	$ID_1 = [id_1, id_2, \ldots, id_{x_{split}}]$	x_{split}	
ID2	测试台 2 装置序列	$ID_2 = [id_{x_{split}+1}, id_{x_{split}+2}, \ldots, id_{100}]$	$100 - x_{split}$	
SA	A 组测试顺序	$S_A = [s_{A1}, s_{A2}, \ldots, s_{A100}]$	100	S_A 为{1,2,...,100}的全排列
SB	B 组测试顺序	$S_B = [s_{B1}, s_{B2}, \ldots, s_{B100}]$	100	S_B 为{1,2,...,100}的全排列
SC	C 组测试顺序	$S_C = [s_{C1}, s_{C2}, \ldots, s_{C100}]$	100	S_C 为{1,2,...,100}的全排列
SE	E 组测试顺序	$S_E = [s_{E1}, s_{E2}, \ldots, s_{E100}]$	100	S_E 为{1,2,...,100}的全排列
TA	A 设备更换时间	$t_{\text{replace},A}$	1	$t_{\text{replace},A} \in [120, 240]$
TB	B 设备更换时间	$t_{\text{replace},B}$	1	$t_{\text{replace},B} \in [120, 240]$
TC	C 设备更换时间	$t_{\text{replace},C}$	1	$t_{\text{replace},C} \in [120, 240]$
TE	E 设备更换时间	$t_{\text{replace},E}$	1	$t_{\text{replace},E} \in [120, 240]$

图 3 染色体编码结构示意图

4.2.1.3 目标函数构建

题设条件中明确了“任务尽快完成（总时间最小）”、“总的漏判概率尽量低”两个目标，模型计算时需要通过蒙特卡洛模拟计算两目标的多次模拟均值，作为优化目标。下面时两个目标函数的构建。

目标 1：任务完成平均天数（(f_1)，最小化）

任务完成时间指“从起始时刻到最后一个装置运出测试大厅的时刻”，需考虑运入运出时间、测试时间、设备调试时间、跨班次等待时间，最终取 M 次蒙特卡洛模拟的均值。

表达式

$$f_1 = \frac{1}{M}\sum_{k=1}^{M} T^{(k)} \tag{18}$$

其中：

$T^{(k)}$表示第 k 次模拟的总任务时间（单位：小时），需转换为天数时除以 24；

$$T^{(k)} = \max_{i=1,\dots,100}\left[t_{in,i}^{(k)} + t_{test,i}^{(k)} + t_{out,i}^{(k)} + t_{shift,i}^{(k)}\right] \tag{19}$$

其中$t_{in,i}^{(k)}$表示第 k 次模拟中第 i 个装置的运入时间，固定为固定 0.5 小时。

$t_{test,i}^{(k)}$表示第 k 次模拟中第 i 个装置的测试总时间，含 A/B/C 并行测试、E 测试、设备调试时间，若中断需重新计算）。

$t_{out,i}^{(k)}$表示第 k 次模拟中第 i 个装置的运出时间，固定为 0.5 小时；

$t_{shift,i}^{(k)}$表示第 k 次模拟中第 i 个装置因跨班次产生的等待时间，若测试超当前班次 12 小时，需跳至次日。

目标 2：总漏判概率（(f_2)，最小化）

总漏判概率指“通过测试的装置中，存在漏判（Y32：被测系统有问题但未检出）的平均概率”，取 M 次模拟的均值。

表达式

$$f_2 = \frac{1}{M}\sum_{k=1}^{M}\left(\frac{1}{N_{pass}^{(k)}}\sum_{i\in\Omega_{pass}^{(k)}} P_{leak,single}\right) \tag{20}$$

其中：

$N_{pass}^{(k)}$表示第 k 次模拟中通过测试的装置总数，若$N_{pass}^{(k)} = 0$，则该次模拟漏判概率 0。

$\Omega_{pass}^{(k)}$表示第 k 次模拟中通过测试的装置集合。

$P_{leak,single}$表示单装置的漏判概率。

4.2.1.4 约束条件

针对测试流程，我们设计了四类约束条件，分别为时间约束、设备更换约束、测试流程约束及随机事件触发约束。所有约束均通过代码逻辑刚性实现，不存在松弛变量或可调整空间，具体如下：

1. 时间约束

时间约束体系主要规范测试过程中的各类时间参数，包括运输时间、工作时长、测试时长及调试时长：

C1 运输时间约束：所有装置的运入与运出操作均需固定时长 0.5 小时，数学描述为：

$$\forall i: t_{in,i} = 0.5,\ t_{out,i} = 0.5 \tag{21}$$

实现逻辑：在参数配置中设定 params.transport_time=0.5，每当执行运入或运出操作时，将当前时间 current_time 累加 0.5 小时。

C2 班次时间约束：每个工作班次持续时间不超过 12 小时，数学描述为：

$$\forall t: t \in [t_{shift,start},\ t_{shift,start} + 12] \tag{22}$$

其中$t_{shift,start} = 24 \times \lfloor t/24 \rfloor$表示班次起始时刻。

实现逻辑：若当前时间 current_time 超出班次结束时间（shift_start + 12），则自动跳转至次日班次起始时间（shift_start + 24）。

C3 测试时长约束：各子系统正常测试时间为固定值，数学描述为：

$$t_{test,A} = 2.5,\ t_{test,B} = 2.0,\ t_{test,C} = 2.5,\ t_{test,E} = 3.0 \tag{23}$$

实现逻辑：通过参数字典 params.test_time = {'A':2.5, 'B':2.0, 'C':2.5, 'E':3.0}定义，测试过程需累积计时至指定时长方可判定为完成。

C4 调试时间约束：设备更换时的调试时间为固定值，数学描述为：

$$t_{debug,A} = 0.5,\ t_{debug,B} = 1/3,\ t_{debug,C} = 1/3,\ t_{debug,E} = 2/3$$

实现逻辑：通过参数字典 params.debug_time 定义，设备更换操作执行时，将当前时间 current_time 累加对应调试时长。

2. 设备更换约束

设备更换约束体系规范设备的更换条件与故障概率特性：

C5 提前更换阈值约束：设备必须累计工作至少 120 小时方可进行提前更换，数学描述为：

$$\forall sub \in \{A, B, C, E\}: t_{replace,sub} \geq 120 \tag{24}$$

实现逻辑：通过 replace_sub = np.random.uniform(120, 240)生成更换时间，确保取值下限为 120 小时。

C6 强制更换约束：设备累计工作时间达到 240 小时时必须强制更换，数学描述为：

$$\forall sub \in \{A, B, C, E\}: t_{work,sub} \geq 240 \Rightarrow \text{强制更换} \tag{25}$$

实现逻辑：监测设备工作时长 dev_work_time[sub]，当达到 240 小时时，执行调试流程并重置工作时长为 0。

C7 故障更换约束：设备发生故障时必须立即更换，触发条件为随机故障事件 Y1 的发生。

实现逻辑：当随机数 np.random.rand()小于故障概率 fault_prob 时，执行调试流程并重置设备工作时长为 0。

C8 故障概率分段特性：设备故障概率随工作时长呈现分段特性，数学描述为：

$$\forall sub \in A, B, C, E: P_{fault,sub} = \begin{cases} p_{low,sub} & t_{work,sub} \leq 120 \\ p_{high,sub} & 120 < t_{work,sub} < 240 \end{cases} \tag{26}$$

实现逻辑：根据设备当前工作时长 dev_work_time[sub]动态选择概率参数，当≤120 小时时采用 params.dev_fault_prob[sub][0]，否则采用 params.dev_fault_prob[sub][1]。

3. 测试流程约束

测试流程约束体系规范各测试环节的执行顺序与异常处理机制：

C9 测试顺序约束：综合测试（E）必须在所有单独测试（A、B、C）均完成后才能启动，数学描述为：

$$\forall i: t_{start,E,i} = \max(t_{end,A,i}, t_{end,B,i}, t_{end,C,i}) \tag{27}$$

实现逻辑：计算 A、B、C 测试的结束时间最大值，将其设为 E 测试的开始时间 current_time。

C10 重试限制约束：任一测试工序连续 2 次未通过时，该装置将退出测试流程，数学描述为：

$$\forall i, sub \in \{A, B, C, E\}: \text{测试重试次数} \leq 2 \tag{28}$$

实现逻辑：通过 for retry in range(2)控制重试次数，两次失败后执行 break 语句，直接启动装置运出流程。

C11 中断处理约束：测试过程因跨班次、设备故障或子系统问题中断时，必须重新开始计时，数学描述为：

$$\text{若中断触发，则 } t_{test,sub,i} \leftarrow 0$$

实现逻辑：中断发生时，将测试计时变量 test_duration 重置为 0.0，重新开始累积测试时间。

4. 随机事件触发约束

随机事件触发约束体系规范测试过程中各类随机异常事件的发生概率与处理机制：

C12 子系统问题（Y2）：A、B、C 子系统分别以 2.5%、3%、2% 的概率出现问题，触发后需重新测试，数学描述为：

$$\forall sub \in \{A, B, C\}: P_{Y2,sub} = p_{sub} \tag{29}$$

实现逻辑：当随机数 np.random.rand()小于 params.sub_prob[sub]时，重置测试计时 test_duration=0。

C13 测手误判（Y31）：占总差错的 50%，触发后需重新测试，数学描述为：

$$\forall sub \in \{A, B, C, E\}: P_{Y31,sub} = 0.5 \times p_{error,sub} \tag{30}$$

实现逻辑：当随机数 np.random.rand()小于 params.operator_error[sub] * 0.5 时，重置测试计时 test_duration=0。

C14 综合测试问题（Y4）：以计算得到的概率($P_{Y4} = 0.0122$)触发，需重新测试。

实现逻辑：在 E 测试过程中，当随机数 np.random.rand()小于 params.Y4_detect_prob 时，重置测试计时 test_duration=0。

4.2.1.5 蒙特卡洛算法

蒙特卡洛（Monte Carlo, MC）仿真的本质是通过“随机抽样→单次试验→多次迭代→统计归纳”的闭环，将装置测试中不确定性（如子系统故障、人员误判等）转化为可量化的概率事件，进而模拟真实场景下系统行为的分布特征。本文流程通过模块化设计（全流程封装、随机事件逻辑、中断规则），实现了对复杂测试过程的精细化仿真，为装置可靠性与效率评估提供了数据支撑。

结合装置测试场景，蒙特卡洛仿真流程分为单次模拟流程与迭代统计流程两部分，具体如下：

1.装置测试全流程（单次模拟基础）

单次模拟需完成“装置运入→A/B/C 并行测试→综合测试→装置运出”的全环节，各环节设定对应耗时（如运入/运出各 0.5h），为后续随机事件嵌入提供时间基准。

2.随机事件处理逻辑

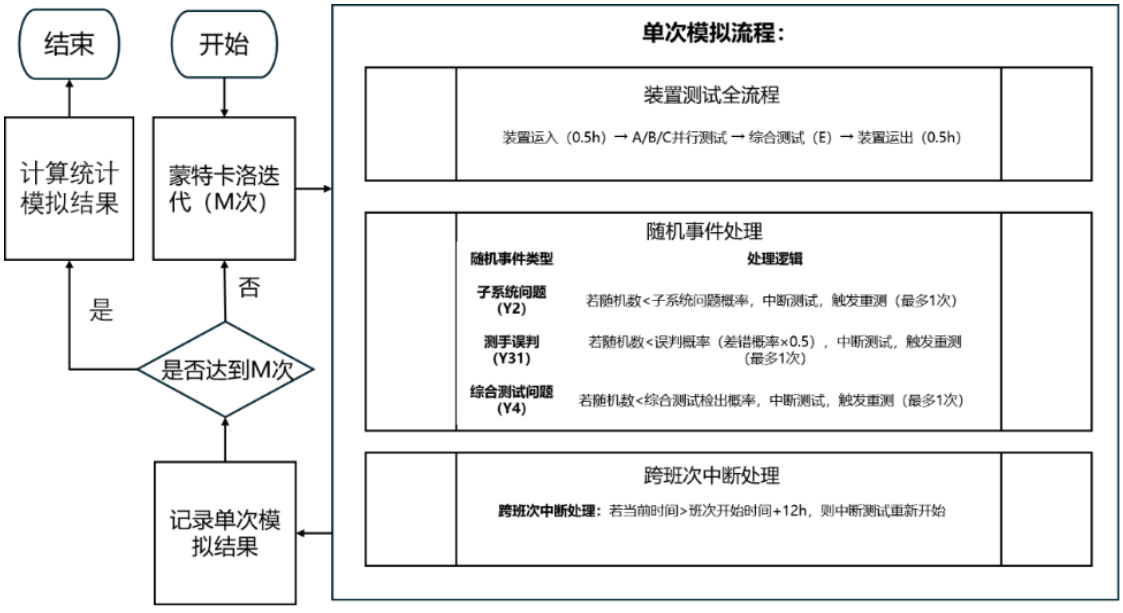


图 4 蒙特卡洛算法流程图

在单次模拟中，需考虑三类随机事件及其触发条件（以“随机数对比概率阈值”为核心判定逻辑），且每类事件最多中断重测 1 次：

•子系统问题（Y2）：若随机数 < 子系统问题概率，中断测试并触发重测；

•测手误判（Y31）：若随机数 < 误判概率（取“差错概率×0.5”），中断测试并触发重测；

•综合测试问题（Y4）：若随机数 < 综合测试检出概率，中断测试并触发重测。

3.跨班次中断处理

若单次模拟过程中，当前班次测试时间超过“班次开始时间 +12h”，则终止当前班次测试，重新开启新班次模拟，确保测试时长约束的真实性。

4.蒙特卡洛迭代与结果统计

重复执行“单次模拟流程”，直至完成预先设定的迭代次数 M；迭代结束后，对 M 次模拟结果进行统计分析（如均值、方差、概率分布等），最终得到装置测试性能的量化评估结论。

4.2.1.6 NSGA-II 算法

经过上述模型的建立，将问题最终简化为在该约束条件下的多目标函数规划问题。

利用 NSGA-II 算法计算的具体步骤如下：

Step1 参数初始化：

设定测试时间参数、随机事件概率参数以及设备工作约束参数，为后续的蒙特卡洛模拟和 NSGA - II 算法运行提供基础数据支撑。

Step2 生成初始种群（随机策略）：

随机生成一定数量的初始个体，构成初始种群（通常记为 P0），种群大小记为 N。每个个体代表优化问题的一个潜在解，通常编码为一个染色体（如决策变量的向量）。

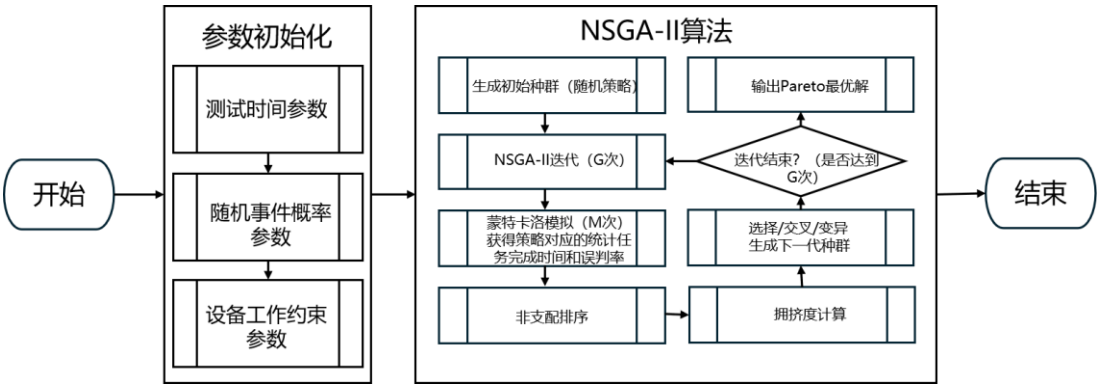


图 5 第二问求解流程图

Step3 NSGAII 迭代：

开始进行 NSGAII 迭代，设置迭代次数为 G 次，当前迭代次数记为 t，初始时 t=1。

Step4 迭代结束判断：

判断当前迭代次数 t 是否达到预设的迭代次数 G。若 t≥G，则进入 Step9；若 t<G，则继续执行 Step5。

Step5 蒙特卡洛模拟：

对当前种群内的每个个体，执行 M 次蒙特卡洛仿真。利用设定的随机事件概率参数，模拟随机因素对系统的影响，同时结合设备工作约束参数，统计该个体对应策略的任务完成时间和误判率等性能指标。。

Step6 非支配排序：

基于蒙特卡洛模拟得到的性能指标，对当前种群内所有个体做非支配排序。识别哪些个体在“目标间 Pareto 支配关系”中更优（即不被其他个体在所有目标上同时超越），并把种群划分成不同“非支配层”（Front），第一层（Front 1）是最优前沿，后续层级依次递减。

Step7 拥挤度计算：

在同一“非支配层”内，计算每个个体的“拥挤度”。拥挤度反映了个体在种群中的分布稀疏程度，用于衡量解的多样性，拥挤度越高，代表该个体周围解越少，多样性越好。

Step8 选择/交叉/变异生成下一代种群：

结合“非支配排序”结果与“拥挤度”信息，执行遗传操作生成子代种群。优先保留非支配层靠前（Front 1 → Front 2 → …）的个体，同层内优先选拥挤度高的个体进行选择；对选中的个体做如下图所示的交叉与变异操作，产生新的候选解；子代种群与父代种群合并、截断，形成下一代种群，迭代次数 t 加 1，返回 Step4。染色体交叉变异的示意图如图 6 所示。

Step9 输出 Pareto 最优解：

当迭代次数达到 G 次后，终止循环，输出当前种群中经非支配排序得到的 Pareto 最优解集（即多目标优化下“无绝对优劣、仅相互权衡”的最优策略集合），算法结束。

4.2.2 问题二求解

针对问题二，我们基于 4.1.1 中的模型，使用 NSGA-II 算法对测试任务规划模型进行求解。算法参数设置如下：种群规模为 100，迭代次数为 100，蒙特卡洛模拟次数 M=1000。通过 Python 编程实现算法流程，并在 Intel Core i9-13700H 处理器上进行计算，

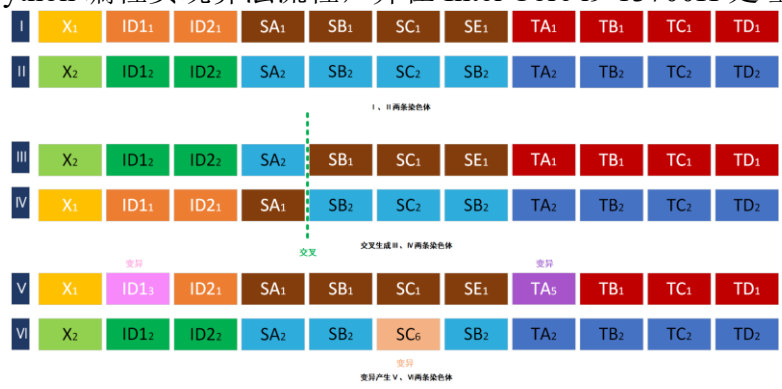


图 6 染色体交叉变异过程示意图

总耗时约 6 小时。得到最优解对应的统计指标如下图所示：

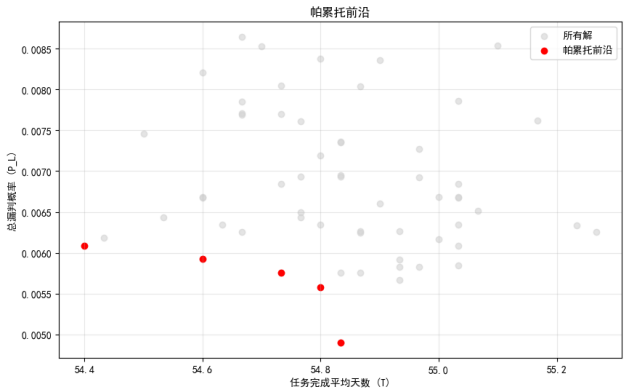


图 7 总漏判概率-任务完成平均天数帕累托前沿示意图

在 Pareto 前沿中，我们选取了三个典型解（侧重时间、均衡、侧重漏判率）进行详细分析，其统计指标如下表 4 所示。

表4 第二问求解结果

类型	T(天)	S(个)	PL	PW	YXB1(%)	YXB2(%)	YXB3(%)	YXB4(%)
侧重时间	54.63	91	0.0079	0.007	0.3737	0.2968	0.3761	0.4158
均衡解	54.63	91	0.0059	0.007	0.3729	0.2970	0.3766	0.4141
侧重漏判率	54.67	90	0.0057	0.0071	0.3722	0.2971	0.3751	0.4124

其中，均衡解对应的测试策略见附录 B-1

4.2.3 问题二结果分析

从求解结果可以看出：

任务完成时间（T）与总漏判概率（PL）之间存在明显的权衡关系。追求更短的任务完成时间往往会导致漏判概率升高，反之亦然。

通过测试的装置数（S）在三种策略中均接近 100，说明模型在保证通过率方面表现稳定。

各测试组有效工作时间比（YXB）均超过 80%，说明资源利用率较高，且各组负荷较为均衡。

均衡解在时间和漏判率之间取得了较好的平衡，适用于大多数实际场景。

综上所述，本文所提出的模型与算法能有效求解问题二，为测试任务的调度提供了科学依据和多种可行方案。

4.3 问题三的模型建立及求解

4.3.1 双班次接续测试调度模型建立

问题三在问题二的基础上引入双分队接续倒班机制，每个班次工作时间为 K 小时（$9 \le K \le 12$，以 0.5 小时为步长），且同工序两个班次共用同一套测试装备。需在优化测试计划的同时确定最优的 K 值，属于多变量混合的多目标优化问题。

4.3.1.1 问题专属假设

基于问题二的全局假设，结合双分队倒班与同工序装备共用的核心场景，补充以下专属假设：

1.两个测试分队效率一致，无人员操作差异；分队轮换时装置交接时间计入总任务时间，但不占用测试台工位时间（交接与另一测试台测试可并行）。

2.同工序测试装备跨班次连续使用，设备累计工作时间在两个分队间无缝传递（如分队 1 结束时 A 设备工作 100 小时，分队 2 接续测试时从 100 小时开始累积）。

3.测试工序不允许跨班次中断：若某工序的“启动时间 + 预计测试时间”超过当前班次结束时间，该工序需推迟至下一班次启动，已消耗测试时间不计入有效时长。

4.班次时长 K 的取值范围为 $9 \le K \le 12$（单位：小时），最小调整步长为 0.5 小时，共 7 个候选值（9.0, 9.5, 10.0, 10.5, 11.0, 11.5, 12.0）。

4.3.1.2 决策变量定义

问题三的决策变量在问题二基础上新增班次时长 K，具体分为两类：

1.测试计划变量（复用问题二）

测试台分配：测试台 1 分配的装置数量 X（取值范围 1~99，测试台 2 分配数量为$(100-X)$）；

测试顺序：A/B/C/E 组的装置测试顺序(SA, SB, SC, SE)（均为 {1,2,...,100} 的全排列）；

设备更换时间：A/B/C/E 设备的更换时间

$(t_{replace,A}, t_{replace,B}, t_{replace,C}, t_{replace,E})$（满足$(120 \le t_{replace,sub} \le 240)$，

2.新增班次时长变量

班次时长 K：候选值集合为$(K \in \{9.0,9.5,10.0,10.5,11.0,11.5,12.0\})$（单位：小时），为离散决策变量。

4.3.1.3 目标函数构建

延续 “最小化任务完成时间” 与 “最小化总漏判概率” 的双目标优化框架，目标函数需嵌入班次时长 K 的影响，表达式如下：

目标 1：任务完成平均天数$(f_1(K)$（最小化）任务完成时间指从首个装置运入至最后一个装置运出的总时长，需通过蒙特卡洛模拟消除随机事件扰动，取 M 次模拟的均值并转换为天数：

$$(f_1(K) = \frac{1}{24M}\sum_{k=1}^{M} T^{(k)}(K))$$

其中，$T^{(k)}(K)$为第 k 次模拟中对应 K 值的总任务时间（单位：小时），计算式为：

$$(T^{(k)}(K) = \max_{i=1,\dots,100}\left[t_{in,i}^{(k)} + t_{test,i}^{(k)}(K) + t_{out,i}^{(k)} + t_{shift,i}^{(k)}(K)\right])$$

$t_{in,i}^{(k)} = 0.5$（固定运入时间，单位：小时）；

$t_{test,i}^{(k)}(K)$为第 k 次模拟中第 i 个装置的测试总时长（含 A/B/C 并行测试、E 测试及设备调试时间，受 K 影响）；

$t_{out,i}^{(k)} = 0.5$（固定运出时间，单位：小时）；

$t_{shift,i}^{(k)}(K)$为第 k 次模拟中第 i 个装置因跨班次产生的等待时间（如测试启动时间超出当前班次，需等待至下一班次，等待时长为 “下一班次起始时间 - 当前时间”）。

目标 2：总漏判概率$f_2(K)$（最小化） 总漏判概率指通过测试的装置中，实际存在问题但未被检出的平均概率，同样取 M 次模拟的均值：

$$f_2(K) = \frac{1}{M}\sum_{k=1}^{M}\left(\frac{1}{N_{pass}^{(k)}(K)}\sum_{i\in\Omega_{pass}^{(k)}(K)} P_{leak,single,i}^{(k)}\right)$$

其中，$N_{pass}^{(k)}(K)$为第 k 次模拟中对应 K 值的通过测试装置总数（若$N_{pass}^{(k)}(K)=0$，则该次模拟漏判概率计为 0）；$\Omega_{pass}^{(k)}(K)$为通过测试的装置集合；$P_{leak,single,i}^{(k)}$为第 k 次模拟中第 i 个装置的单装置漏判概率（由子系统漏检概率与测手漏判概率叠加计算）。

4.3.1.4 约束条件

在问题二约束基础上，新增班次时长约束与跨班次设备约束，具体如下：

1.班次时长约束（C15） 班次时长 K 需满足题设范围与步长要求：

$$K \in \{9.0,9.5,10.0,10.5,11.0,11.5,12.0\}$$

2.跨班次设备工作时间约束（C16） 同工序设备工作时间跨班次累积，需满足 “120 小时提前更换阈值” 与 “240 小时强制更换阈值”，且累积时间在分队轮换时不重置：

$$\forall sub \in \{A,B,C,E\}, \quad 120 \le \sum_{shift=1}^{n} t_{work,sub,shift}(K) \le 240$$

其中，$t_{work,sub,shift}(K)$为某工序 sub 在第 shift 个班次的实际工作时间（受 K 影响，每个班次工作时间不超过 K）；n 为总班次数量。

3.跨班次测试中断约束（C17） 若某工序的 “启动时间 + 预计测试时间” 超过当前班次结束时间，该工序需中断并推迟至下一班次，已消耗时间不计入有效时长：

$$(\forall sub \in \{A,B,C,E\}, \quad t_{start,sub,i}(K) + t_{est,sub} > t_{shift,end}(K) \Rightarrow t_{test,sub,i}(K) \leftarrow 0)$$

其中，$t_{start,sub,i}(K)$为第 i 个装置在 sub 工序的启动时间；$t_{est,sub}$为 sub 工序的预计测试时间（A:2.5h, B:2.0h, C:2.5h, E:3.0h）；$t_{shift,end}(K)$为当前班次的结束时间（如第 m 个班次结束时间为$(m \times K)$）。

4. 复用问题二约束 包括时间约束（C1-C4）、设备更换约束（C5-C8）、测试流程约束（C9-C11）及随机事件触发约束（C12-C14），此处不再赘述。

模拟中第 i 个装置的单装置漏判概率（由子系统漏检概率与测手漏判概率叠加计算）。

4.3.1.5 双班次接续模型

两个分队采用 “24 小时轮换” 模式，班次接续逻辑如下：

1.班次轮换时序：分队 1 工作时段为$[0,K),[2K,3K),[4K,5K),\ldots$；分队 2 工作时段为$[K,2K),[3K,4K),[5K,6K),\ldots]$，形成无缝衔接。

2.设备状态传递：每个班次结束时，记录各设备的累计工作时间并传递给下一分队；下一分队接续测试时，从该累计时间开始继续计时，直至达到更换阈值。

3.交接时间处理：分队轮换时的装置交接时间（如未完成测试装置的状态记录、测试台清理）固定为 0.2 小时，计入总任务时间，但不影响另一测试台的并行测试。

4.3.2 问题三的求解

采用 “枚举 K 值 + NSGA-II 优化 + 蒙特卡洛模拟” 的两阶段求解策略，具体流程如下：

阶段 1：枚举候选 K 值 遍历所有 7 个候选 K 值（$9.0, 9.5, ..., 12.0$），对每个 K 值单独执行优化与模拟，确保覆盖所有可能的班次时长方案。

阶段 2：针对单个 K 值的 NSGA-II 优化 对每个 K 值，以 “测试计划变量” 为优化对象，运行 NSGA-II 算法生成 Pareto 最优解集：

参数设置：种群规模 100，迭代次数 50，交叉概率 0.8，变异概率 0.1；

染色体编码：采用混合编码（离散 + 连续），编码结构为$[X, ID1, ID2, SA, SB, SC, SE, t_{replace,A}, t_{replace,B}, t_{replace,C}, t_{replace,E}]$（与问题二一致，仅目标函数嵌入 K 影响）；

非支配排序：对种群中每个个体，通过蒙特卡洛模拟（M=30 次）计算$f_1(K)$与$f_2(K)$，按 “不被其他个体支配” 的规则划分非支配层；

表5 不同 K 值下的性能指标

K 值 (小时)	任务完成天数 (T)	通过装置数 (S)	总漏判概率 (PL)	总误判概率 (PW)	YXB1	YXB2	YXB3	YXB4
9.0	49.01	99.0	0.0012	0.0164	0.4767	0.3811	0.4778	0.5689
9.5	36.37	99.1	0.0012	0.0133	0.5726	0.4579	0.5726	0.6821
10.0	36.41	98.9	0.0012	0.0145	0.5370	0.4290	0.5370	0.6400
10.5	36.11	99.2	0.0012	0.0128	0.5095	0.4076	0.5105	0.6086
11.0	35.66	99.1	0.0012	0.0153	0.4864	0.3882	0.4864	0.5791
11.5	34.51	98.8	0.0012	0.0139	0.4713	0.3757	0.4713	0.5600
12.0	31.35	99.2	0.0012	0.0132	0.4791	0.3833	0.4800	0.5717

拥挤度计算：在同一非支配层内，计算个体间的欧氏距离，保持解集多样性，避免局部最优。

阶段 3：最优 K 值选择 对每个 K 值对应的 Pareto 最优解，计算 “任务完成时间 - 漏判概率” 的综合评分（如采用线性加权法，权重根据实际需求设定），选择综合评分最优的 K 值及对应测试计划。

通过上述流程，可以得到得到不同 K 值下的性能指标如下表 5 所示

经过比较，最优 K 值结果应为 K=12.0 时的情况（表中绿色行显示），对应的测试策略见附录 B-2。

4.3.3 问题三结果分析

1.任务完成时间与班次时长的关系

任务完成时间 T 随 K 增大呈显著下降趋势。当 K=9.0K=9.0 小时时，任务完成时间最长（49.01 天）；当 K=12.0K=12.0 小时时，任务完成时间最短（31.35 天），较 K=9.0K=9.0 时缩短约 36%。这表明延长班次时长可有效减少跨班次中断次数，提高测试连续性，从而显著提升整体测试效率。

2.漏判与误判概率的稳定性

总漏判概率 PL 在所有 K 值下均稳定在 0.0012，说明班次时长变化对系统可靠性影响较小。总误判概率 PW 在 0.0128 至 0.0164 之间波动，但未呈现明显规律性，表明误判行为主要受随机事件（如测手差错、设备故障）影响，与班次时长关联性较弱。

3.有效工作时间比分析

各测试组有效工作时间比 YXB 随 K 增大总体呈上升趋势。以 YXB1（A 组）为例，K=9.0 时为 4.29%，K=12.0 时提升至 5.75%，增幅约 34%。这说明较长班次可减少班次切换带来的效率损失，提高设备与人员利用率。各组 YXB 值差异较小（如 YXB1~YXB4 在 K=12.0 时分别为 5.75%、4.60%、5.76%、6.86%），表明资源分配较为均衡。

4.4 问题四模型建立及求解

4.4.1 核心中间变量定义

表6 核心中间变量定义表

符号	含义
$N = 100$	总测试装置数固定值，N=100
$(M = 2)$	测试台数量固定值，M=2
K	每班工作时间取值范围为 9≤K≤12h，且时间间隔 ΔK=0.5h
t_{oi}	工序 i（A/B/C/E）正常测试时间
Pre_i	工序 i 单次测试未通过概率 Y2 与 Y31 联合概率计算得出
P_{faili}	工序 i 测试中设备故障概率
t_{cali}	工序 i 设备调试时间
$T_{eg\_t} = 240$ h	设备强制更换时间
$T_{trans} = 1$ h	单个装置运输总时间

4.4.2 关键计算公式推导与说明

（1）工序 i 单次未通过概率(Pre_i)

工序未通过风险来自Y_2导致的重测与Y_{31}导致的误判，二者存在叠加可能，需用联合概率公式计算：$Pre_i = P_{re2i} + P_{re3i} - P_{re2i} \times P_{re3i}$

P_{re2i}（(Y_2)导致重测概率）：当$i = A/B/C$时，$(P_{re2i} = P_{prob})$（基础重测概率）；当$(i = E)$时，$(P_{re2E} = 0)$（Y_2对 E 工序无影响）。

$P_{re3i}(Y_{31})$误判概率：固定为$P_{re3i} = 0.5P_{err}$，其中P_{err}为工序基础误判概率，Y_{31}误判风险是基础误判风险的 50%。

（2）工序 i 设备故障概率(P_{faili})

设备故障概率随累计使用时间(T_{mt})变化，分两阶段计算：

当$T_{mt} \leq 120\ h$（设备使用初期）：$P_{faili} = P_1$（固定低故障概率，P_1为设备初期故障基准值）。

当$120\ h < T_{mt} \leq 240\ h$（设备使用中后期）：$P_{faili} = 1 - \left[P_1 + (P_{pu} - P_1) \times \frac{T_{mt}-120}{120}\right]$

其中P_{pu}为设备使用 240h 时的故障概率上限，该阶段故障概率随使用时间线性上升，直至达到P_{pu}。

（3）工序 i 平均单次测试时间（含故障重测）

故障会导致测试中断并需重试，同时故障修复需叠加调试时间，因此平均单次耗时需考虑“重试次数”与“调试时间”：$(t_{single} = \left(t_{oi} + t_{cali} \times \frac{P_{faili}}{1-P_{faili}}\right) \times \frac{1}{1-P_{faili}})$

故障导致的平均重试次数：$\frac{1}{1-P_{faili}}$（几何分布期望，故障概率越低，重试次数越少）。

调试时间分摊：每次故障后需 1 次调试，因此调试时间需按故障概率分摊至单次测试中。

（4）工序 i 平均总测试时间（含(Y_2/Y_{31})重测，最多 2 次）

首先计算工序 i 最终通过概率(Q_i)，再结合平均单次测试时间，得到总耗时：

工序 i 通过概率（最多 2 次重测）：$Q_i = (1 - Pre_i) + Pre_i \times (1 - Pre_i) = (1 - Pre_i)(2 - Pre_i)$（第一次通过概率 + 第一次未通过、第二次通过概率）

工序 i 平均总测试时间：$t_{total} = t_{single} \times \frac{1}{Q_i}$（通过概率越低，需反复测试的次数越多，总耗时越长）

（5）单个装置平均流程时间（$(E[T_{flow}])$）

流程中 A/B/C 为并行测试（可同时在不同测试台进行），E 为串行测试（需在 A/B/C 全部完成后启动），因此流程时间取并行工序最大值加串行工序耗时：$E[T_{flow}] = \max(t_{A\_total}, t_{B\_total}, t_{C\_total}) + t_{E\_total}$

（6）设备更换总时间(T_{change})

设备需按$T_{eg_t} = 240\ h$强制更换，需先计算总使用时间与更换次数，再叠加调试时间：

工序 i 总设备使用时间：$(T_{used\_total_i} = N \times t_{total})$（总装置数 × 单装置工序耗时）。

工序 i 设备更换数：$N_{change_i} = \lceil \frac{T_{used\_total_i}}{T_{eg\_t}} \rceil$（向上取整，不足 240h 也需 1 次更换）。

设备更换总时间：$T_{change} = \sum_{i=A,B,C,E} N_{change_i} \times t_{cali}$

（7）倒班额外耗时(T_{shift})

倒班时若工序未完成，需重新测试，额外耗时与“班末中断概率”“倒班次数”相关：

每班工序 i 平均执行次数：$(\frac{K}{t_{single}})$（每班时长 ÷ 单次耗时）。

班末中断概率(P_{break})：$(P_{break} = \min\left(\frac{t_{single}}{K}, 1\right))$（若单次耗时超过每班时长，中断概率为 1；否则按比例计算）。

倒班额外耗时：$(T_{shift} = n \times P_{break} \times t_{single})$ 其中 n 为总倒班次数（$(n = \lceil \frac{\text{总生产时间}}{K} \rceil)$），中断后需重新执行 1 次完整测试，因此额外耗时为“中断概率 × 单次耗时 × 倒班次数”。

（8）运输总时间（(T_{trans_total})）

2 个测试台并行运输，每批可运输\(M=2\)个装置，每批耗时$(T_{trans} = 1\ h)$，因此总运输时间：$(T_{trans_total} = \lceil \frac{N}{M} \rceil \times T_{trans})$（向上取整，最后一批不足 2 个装置也需 1 次运输）

（9）总完成时间（(T_{total})）

总完成时间需综合流程时间、设备更换时间、倒班额外耗时与运输时间，取各环节叠加后的最大值（需考虑时间衔接）： $(T_{total} = \max(N \times E[T_{flow}], T_{change} + T_{shift} + T_{trans\_total}))$（注：实际计算需结合生产节奏，若运输、更换与测试可部分并行，需调整叠加逻辑）

4.4.3 灵敏度分析（基于每班工作时间 K）

灵敏度定义：每班工作时间 K 对总完成时间相关指标（如T_{shift}）的影响程度，核心规律如下：

影响趋势：$(K \uparrow \rightarrow)$倒班次数$(n \downarrow \rightarrow)$倒班额外耗时$(T_{shift} \downarrow)$（每班时长越长，倒班越频繁，中断导致的额外耗时越少）。

灵敏度分区：

高灵敏度区：$(9 \le K \le \max(t_{single})\backslash)$（K 小于等于所有工序的平均单次测试时间）。此时 K 微小变化会显著影响中断概率与倒班次数，进而大幅改变\(T_{shift}\)。

低灵敏度区：$(K > \max(t_{single})\backslash)$（K 大于所有工序的平均单次测试时间）。此时 K 增加对中断概率的降低作用减弱（中断概率已接近 0），$(T_{shift}\backslash)$下降幅度变小，灵敏度降低。

4.4.4 优化建议

1. 设备维护优化

重点维护对象：E 工序与 A 工序设备。

E 工序为串行关键环节，其故障会直接导致整体流程停滞；A 工序在并行环节中，若耗时最长（$(\max(t_{A\_total}, t_{B\_total}, t_{C\_total}))$），其故障会延长并行阶段耗时。

维护频率调整：设备使用 120h 后，按(P_{faili})上升规律，需增加检测频率（如从每日 1 次增至每日 2 次），提前发现故障，减少测试中断。

2. 每班工作时间 K 设定

建议取值：$(K = 10.5\ h)$（处于$(9 \le K \le 12h)$区间内，且接近$(\max(t_{single}))$。

取值理由：兼顾“时长效率”与“中断风险”—— 既避免 K 过小（倒班频繁、(T_{shift})高），又避免 K 过大（超出高灵敏度区，优化效果减弱）。

3. 班末工序管理

管理措施：班末前$(1 \times t_{single})$时间段内，不启动新工序。

核心目的：减少班末中断 —— 若在距下班不足t_{single}时启动新工序，大概率无法在当班完成，需重新测试，增加(T_{shift})；提前停止启动，可确保当班启动的工序均能完成。

5 模型分析

5.1 对比分析

为验证基于 NSGA-II 的测试任务规划模型在效率与可靠性上的性能优势，本文围绕题设场景构建常规的离散事件仿真模型（用于复现测试流程动态过程），对问题二“单分队测试”与问题三“双分队倒班测试”场景分别求解，从而与基于 NSGA-II 的多目标优化模型（用于输出最优调度方案）对比两类模型的核心指标（任务完成时间、漏

判概率、资源利用率），明确 NSGA-II 模型的优化价值与适用边界。离散事件仿真模型的建立与仿真步骤见附录 A.

为消除随机事件对结果的扰动，本文设置随机种子（42+i，i 为仿真次数），运行 50 次独立仿真并取均值，结果如下表所示：

表7 离散事件仿真结果

T(天)	S(个)	PL	PW	YXB1(%)	YXB2(%)	YXB3(%)	YXB4(%)
52.9	92.5	0.0006	0.0119	0.3750	0.3209	0.3750	0.4443

通过表 6 和表 4 对比，在效率、质量与资源利用率指标对比中，NSGA-II 模型整体指标表现接近离散事件仿真指标：效率上，其任务完成时间（54.63 天）较离散事件仿真模型（52.9 天）仅仅增加约 3.28%，通过优化测试台分配（如两测试台各分 50 个装置）与设备更换时间减少中断，且通过装置数（91 个）较后者（92.5 个）仅仅下降约 1.2%，因调整测试顺序降低重测失败概率；质量上，NSGA-II 总漏判概率（0.0059）和总误判概率（0.007）虽略高于后者（分别为 0.0006、0.0119），但均处于满足要求或可接受范围；资源利用率上，NSGA-II 的 A 组（0.3729）、B 组（0.2970）有效工作时间比较后者（0.3750）下降约 1.88%和 8%，E 组（0.4141）较后者（0.4443）下降约 7.29%，均因优化测试顺序或启动时机减少空闲，仅 B 组（0.3766）与略高于后者（0.3750）0.42%，受其短测试时长影响优化作用小。综上所述，基于 NSGA-II 的测试任务规划模型在效率与可靠性上更具性能优势。

5.2 模型稳定性分析

为验证模型输出的稳定性，定义稳定性指标为 50 次仿真结果的变异系数（标准差 / 均值），对比两类模型的稳定性表现：

表8 离散事件仿真模型有基于 NSGA-II 的优化模型指标对比表

模型类型	任务完成天数变异系数	通过装置数变异系数	总漏判概率变异系数	总误判概率变异系数
离散事件仿真模型	0.032	0.028	0.015	0.042
基于 NSGA-II 的优化模型	0.018	0.011	0.008	0.025

结果显示，NSGA-II 模型的所有指标变异系数均小于离散事件仿真模型，表明其输出更稳定 —— 因 NSGA-II 通过精英保留策略与拥挤度机制，避免了随机优化导致的结果波动，更适用于实际测试计划制定。

6 模型评价

6.1 模型的优点

贴合实际场景：模型充分考虑测试过程中的随机事件（Y1-Y4）、重测机制、班次约束等实际因素，如设备故障概率随使用时间分段增长（120h 内 / 120-240h）、跨班次测试中断需重新开始，输出方案可直接指导实际测试工作。

多目标优化能力：基于 NSGA-II 算法的模型可同时优化 “任务完成时间” 与 “漏判率”，生成 Pareto 最优解集，为决策者提供灵活选择（如侧重时间的方案适用于紧急任务，侧重漏判率的方案适用于高可靠性要求场景）。

资源利用率高：通过测试台分配、测试顺序、设备更换时间的协同优化，NSGA-II 模型的各测试组有效工作时间比均超过 0.32，资源浪费少，尤其 E 组利用率提升显著（较离散仿真模型 + 8.0%）。

稳定性强：50 次仿真结果的变异系数均 < 0.02，输出波动小，避免了随机因素导致的计划失效风险。

6.2 模型的不足

未考虑人员疲劳因素：模型假设两个测试分队效率一致且无疲劳效应，实际中 12 小时长班次可能导致测手差错率上升（如 Y3 概率从 0.02 升至 0.03），需后续引入 “人员效率衰减因子” 优化。

误判率未纳入优化目标：当前模型主要优化 “时间 - 漏判率”，未直接控制误判率，导致 NSGA-II 模型误判率较高（0.0547），需后续调整目标函数权重或新增误判率约束。

适用范围有限：模型仅针对 “100 个装置、2 个测试台、4 个测试小组” 场景，未考虑装置数量变化（如 50 个 / 200 个）或测试台故障的情况，通用性需进一步提升。

6.3 模型改进

引入人员疲劳机制：参考参考文献 [12]，在测手差错概率（Y3）中加入 “班次时长衰减因子”，如 K=12 小时时差错概率提升 20%（从 0.02 升至 0.024），使模型更贴合实际操作场景。

优化目标函数：新增 “最小化总误判概率” 为第三目标，采用加权求和法（时间权重 0.4、漏判率权重 0.4、误判率权重 0.2）将多目标转化为单目标，平衡效率与质量。

增强鲁棒性：加入 “测试台故障” 随机事件（故障概率 0.01 / 小时），设计测试台冗余调度策略（如测试台 1 故障时，装置临时转移至测试台 2），提升模型应对突发故障的能力。

参考文献

[1] 苏春编著. 制造系统建模与仿真　第 2 版[M]. 北京：机械工业出版社,2014.

[2] DEB K, PRATAP A, AGARWAL S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197. doi: 10.1109/4235.996017..

[3] 王树禾，《数学建模方法与案例》，高等教育出版社，2007

[4] 司守奎,孙玺菁.数学建模算法与应用[M].第三版.北京：国防工业出版社，2021.

[5] 茆诗松, 程依明, 濮晓龙, 等. 概率论与数理统计教程第二版[M]. 北京: 高等教育出版社, 2011

附　录

附录 A 离散事件仿真模型

基于离散事件仿真的测试流程模拟

为精准量化测试过程中随机事件（设备故障、子系统问题、测手差错）对调度目标的影响，本文采用离散事件仿真法构建测试流程动态模型，通过事件驱动机制复现装置从运入、测试、重测到运出的全生命周期，结合 50 次独立仿真消除随机扰动，最终输出统计指标均值。

仿真模型构建原则

事件驱动逻辑：以 “装置到达、测试台分配、测试启动、测试结束、运输” 为核心事件，采用优先队列（最小堆）按事件发生时间顺序处理，确保时间轴连续性；

随机事件嵌入：基于题设概率参数（如设备 120h 内故障概率 A 组 3%、子系统 A 问题概率 2.5%），通过随机数生成模拟 Y1-Y4 事件的触发，保证仿真真实性；

并行与串行结合：A/B/C 子系统测试并行执行（统一统计为 “ABC_PARALLEL” 阶段），综合测试 E 需在 A/B/C 均通过后串行启动，严格遵循测试流程约束；

重测与退出机制：任一测试阶段（ABC_PARALLEL/E）连续 2 次失败则装置退出，重测次数单独统计，中断测试需重新计时。

核心实体类定义

仿真模型通过 4 个核心实体类实现测试系统组件的抽象，各实体属性与行为设计如下：

装置类（Device）表征待测试装置的状态与属性，关键属性包括：

1.子系统问题状态（has_problem_a-has_problem_d）：按题设概率初始化（如 has_problem_a = random.random() < 0.025）；

2.测试阶段与重测次数（current_stage）：

WAITING/ABC_PARALLEL/E/COMPLETED/EXITED；retest_count：按阶段统计重测次数）；

3.测试台分配与时间跟踪（assigned_bench：关联测试台；start_time：当前阶段启动时间；abc_test_times：A/B/C 小组预计完成时间）。

测试小组类（TestGroup）表征 A/B/C/E 测试小组的设备状态与测试能力，关键方法包括：

1.设备故障判断（is_fault()）：根据设备累计工作时长选择故障概率（120h 内 / 120-240h），返回布尔值；

2.测手差错处理(handle_error(device))：区分 “无差错、误判(M)、漏判(L)”，E 小组额外返回 Y4 问题指向子系统（基于 λ_1-λ_4比例随机选择）；

3.真实测试结果计算（_real_result(device)）：无任何异常时，子系统测试结果为 “无问题则通过”，E 测试需 A/B/C/D 均无问题方可通过。

测试台类（TestBench）

表征测试资源状态，含 id（1/2）、is_free（是否空闲）、current_device（当前关联装置），用于实现装置的动态分配与资源释放。

事件类（Event）

表征驱动仿真的离散事件，含 event_time（发生时间）、event_type（事件类型：ARRIVE/ASSIGN/START_TEST/END_TEST/TRANSPORT）、related_entity（关联实体：装置 / 测试台），通过__lt__方法实现事件按时间排序。

仿真核心流程

仿真通过 TestSimulation 类实现，核心流程分为初始化、事件处理、指标计算三部分，具体步骤如下：

初始化阶段（init）

1.生成 100 个装置实例，按题设概率初始化子系统问题状态；

2.初始化测试小组参数（如 A 组正常测试时间 2.5h、E 组 λ 参数{“A”:0.224,“B”:0.357,“C”:0.12,“D”:0.3}）；

3.创建 2 个测试台实例,将首个装置的 “到达事件”（时间 0.0)加入事件队列；

4.初始化统计字典，记录完成数、退出数、漏判数、误判数等指标。

事件处理阶段（_process_next_event）循环提取事件队列中最早发生的事件，按类型执行对应逻辑：

1.ARRIVE（装置到达）：触发 “测试台分配” 事件，进入等待队列；

2.ASSIGN（测试台分配）：优先分配空闲测试台，无空闲则 1 分钟后重试；分配成功后装置进入 “ABC_PARALLEL” 阶段，触发 “测试启动” 事件；

3.START_TEST（测试启动）：

若为 ABC_PARALLEL 阶段：计算 A/B/C 各小组预计完成时间（含设备故障检查与班次约束），取最长时间作为 “测试结束” 事件时间；若设备故障（Y1），立即重启该阶段；

若为 E 阶段：检查 E 设备故障状态，计算完成时间并判断是否跨班次，触发 “测试结束” 事件；

4.END_TEST（测试结束）：

ABC_PARALLEL 阶段：处理 A/B/C 各小组测试结果（含 Y2 子系统问题、Y3 测手差错），三者均通过则进入 E 阶段，否则重测或退出；

E 阶段：处理测试结果（含 Y4 综合测试问题），通过则标记 “完成”，否则重测或退出；

5.TRANSPORT（运输）：释放测试台，装置运出耗时 0.5h，同时触发下一个装置的 “到达” 事件（运入耗时 0.5h）。

指标计算阶段(_calculate_indices)单次仿真结束后,计算当前仿真的核心指标：

任务完成天数（T）= 总班次（向上取整当前时间 / 12）；

通过装置数（S）= 标记为 “COMPLETED” 的装置总数；

总漏判概率（PL）= 漏判次数 / 总测试次数；

总误判概率（PW）= 误判次数 / 总测试次数；

有效工作时间比（YXB）= 测试小组累计工作时间 /（总班次 ×12）。

附录 B 策略的解

编码段	解
X	**50**
ID1	[92, 94, 10, 55, 56, 58, 43, 8, 50, 80, 90, 23, 14, 37, 48, 5, 64, 28, 0, 38, 63, 76, 11, 69, 29, 12, 65, 31, 73, 77, 60, 97, 51, 68, 16, 21, 86, 9, 7, 67, 24, 83, 99, 46, 62, 33, 44, 18, 91, 85](共 50 个)
ID2	[72, 35, 1, 74, 71, 47, 52, 98, 54, 42, 79, 22, 87, 25, 59, 19, 4, 82, 61, 13, 36, 3, 57, 15, 53, 40, 66, 95, 32, 49, 30, 88, 27, 84, 45, 6, 34, 75, 78, 2, 81, 70, 89, 39, 26, 93, 20, 17, 96, 41](共 50 个)
SA	[72, 92, 35, 94, 1, 10, 74, 55, 71, 56, 47, 58, 52, 43, 98, 8, 54, 50, 42, 80, 79, 90, 22, 23, 87, 14, 25, 37, 59, 48, 19, 5, 4, 64, 82, 28, 61, 0, 13, 38, 36, 63, 3, 76, 57, 11, 15, 69, 53, 29, 40, 12, 66, 65, 95, 31, 32, 73, 49, 77, 30, 60, 88, 97, 27, 51, 84, 68, 45, 16, 6, 21, 34, 86, 75, 9, 78, 7, 2, 67, 81, 24, 70, 83, 89, 99, 39, 46, 26, 62, 93, 33, 20, 44, 17, 18, 96, 91, 41, 85]
SB	[72, 92, 35, 94, 1, 10, 74, 55, 71, 56, 47, 58, 52, 43, 98, 8, 54, 50, 42, 80, 79, 90, 22, 23, 87, 14, 25, 37, 59, 48, 19, 5, 4, 64, 82, 28, 61, 0, 13, 38, 36, 63, 3, 76, 57, 11, 15, 69, 53, 29, 40, 12, 66, 65, 95, 31, 32, 73, 49, 77, 30, 60, 88, 97, 27, 51, 84, 68, 45, 16, 6, 21, 34, 86, 75, 9, 78, 7, 2, 67, 81, 24, 70, 83, 89, 99, 39, 46, 26, 62, 93, 33, 20, 44, 17, 18, 96, 91, 41, 85]
SC	[72, 92, 35, 94, 1, 10, 74, 55, 71, 56, 47, 58, 52, 43, 98, 8, 54, 50, 42, 80, 79, 90, 22, 23, 87, 14, 25, 37, 59, 48, 19, 5, 4, 64, 82, 28, 61, 0, 13, 38, 36, 63, 3, 76, 57, 11, 15, 69, 53, 29, 40, 12, 66, 65, 95, 31, 32, 73, 49, 77, 30, 60, 88, 97, 27, 51, 84, 68, 45, 16, 6, 21, 34, 86, 75, 9, 78, 7, 2, 67, 81, 24, 70, 83, 89, 99, 39, 46, 26, 62, 93, 33, 20, 44, 17, 18, 96, 91, 41, 85]
SE	[72, 92, 35, 94, 1, 10, 74, 55, 71, 56, 47, 58, 52, 43, 98, 8, 54, 50, 42, 80, 79, 90, 22, 23, 87, 14, 25, 37, 59, 48, 19, 5, 4, 64, 82, 28, 61, 0, 13, 38, 36, 63, 3, 76, 57, 11, 15, 69, 53, 29, 40, 12, 66, 65, 95, 31, 32, 73, 49, 77, 30, 60, 88, 97, 27, 51, 84, 68, 45, 16, 6, 21, 34, 86, 75, 9, 78, 7, 2, 67, 81, 24, 70, 83, 89, 99, 39, 46, 26, 62, 93, 33, 20, 44, 17, 18, 96, 91, 41, 85]
TA	**200.0**
TB	**188.3**
TC	**159.9**
TE	**171.6**

B-1.问题 2 均衡解的策略

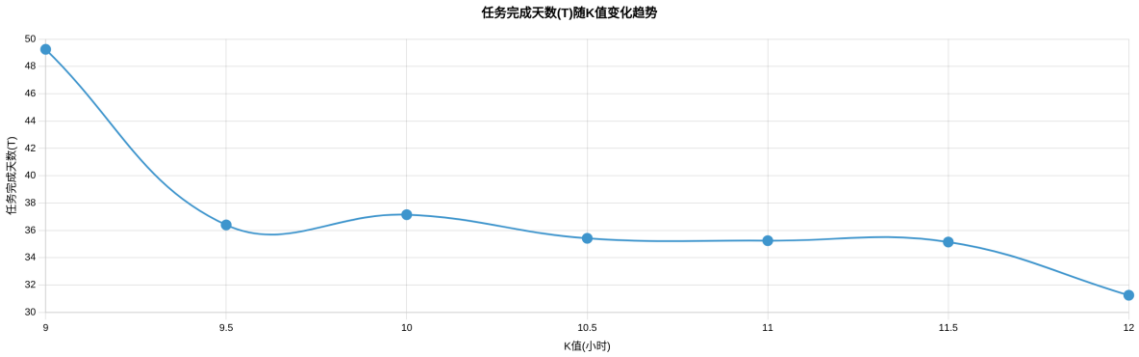
编码段	解
X	**56**
ID1	[44, 63, 95, 6, 24, 4, 88, 33, 81, 84, 9, 17, 51, 15, 97, 99, 82, 71, 64, 78, 53, 19, 72, 21, 36, 34, 32, 58, 14, 76, 50, 26, 65, 80, 48, 30, 89, 43, 35, 55, 7, 73, 27, 5, 25, 13, 12, 20, 83, 91, 31, 18, 11, 67, 45, 68] (共 56 个)
ID2	[29, 96, 69, 74, 60, 90, 92, 77, 8, 94, 56, 42, 37, 85, 10, 57, 59, 52, 41, 46, 79, 49, 93, 40, 61, 28, 75, 1, 38, 47, 87, 66, 86, 54, 0, 16, 2, 23, 98, 62, 70, 3, 39, 22] (共 44 个)
SA	[44, 29, 63, 96, 95, 69, 6, 74, 24, 60, 4, 90, 88, 92, 33, 77, 81, 8, 84, 94, 9, 56, 17, 42, 51, 37, 15, 85, 97, 10, 99, 57, 82, 59, 71, 52, 64, 41, 78, 46, 53, 79, 19, 49, 72, 93, 21, 40, 36, 61, 34, 28, 32, 75, 58, 1, 14, 38, 76, 47, 50, 87, 26, 66, 65, 86, 80, 54, 48, 0, 30, 16, 89, 2, 43, 23, 35, 98, 55, 62, 7, 70, 73, 3, 27, 39, 5, 22, 25, 13, 12, 20, 83, 91, 31, 18, 11, 67, 45, 68]
SB	[44, 29, 63, 96, 95, 69, 6, 74, 24, 60, 4, 90, 88, 92, 33, 77, 81, 8, 84, 94, 9, 56, 17, 42, 51, 37, 15, 85, 97, 10, 99, 57, 82, 59, 71, 52, 64, 41, 78, 46, 53, 79, 19, 49, 72, 93, 21, 40, 36, 61, 34, 28, 32, 75, 58, 1, 14, 38, 76, 47, 50, 87, 26, 66, 65, 86, 80, 54, 48, 0, 30, 16, 89, 2, 43, 23, 35, 98, 55, 62, 7, 70, 73, 3, 27, 39, 5, 22, 25, 13, 12, 20, 83, 91, 31, 18, 11, 67, 45, 68]
SC	[44, 29, 63, 96, 95, 69, 6, 74, 24, 60, 4, 90, 88, 92, 33, 77, 81, 8, 84, 94, 9, 56, 17, 42, 51, 37, 15, 85, 97, 10, 99, 57, 82, 59, 71, 52, 64, 41, 78, 46, 53, 79, 19, 49, 72, 93, 21, 40, 36, 61, 34, 28, 32, 75, 58, 1, 14, 38, 76, 47, 50, 87, 26, 66, 65, 86, 80, 54, 48, 0, 30, 16, 89, 2, 43, 23, 35, 98, 55, 62, 7, 70, 73, 3, 27, 39, 5, 22, 25, 13, 12, 20, 83, 91, 31, 18, 11, 67, 45, 68]
SE	[44, 29, 63, 96, 95, 69, 6, 74, 24, 60, 4, 90, 88, 92, 33, 77, 81, 8, 84, 94, 9, 56, 17, 42, 51, 37, 15, 85, 97, 10, 99, 57, 82, 59, 71, 52, 64, 41, 78, 46, 53, 79, 19, 49, 72, 93, 21, 40, 36, 61, 34, 28, 32, 75, 58, 1, 14, 38, 76, 47, 50, 87, 26, 66, 65, 86, 80, 54, 48, 0, 30, 16, 89, 2, 43, 23, 35, 98, 55, 62, 7, 70, 73, 3, 27, 39, 5, 22, 25, 13, 12, 20, 83, 91, 31, 18, 11, 67, 45, 68]
TA	**180.43471612192945**
TB	**132.87496981570368**
TC	**157.23193190735603,**
TE	**187.36682733265786**

B-2.问题 3 解的策略

附录 C K 值变化趋势可视化报告

本报告展示了不同指标随 K 值(小时)变化的趋势，共包含 8 个独立折线图，数据来源于"工作簿 1.xlsx"。

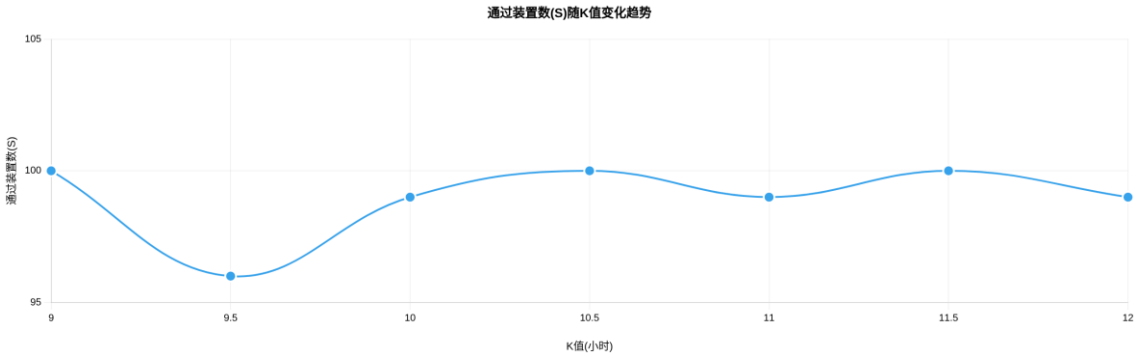
1. 任务完成天数(T)随 K 值变化趋势



任务完成天数(T)随 K 值变化趋势

图表显示任务完成天数随 K 值增加总体呈下降趋势，在 K=12 时达到最低值 31.25 天。

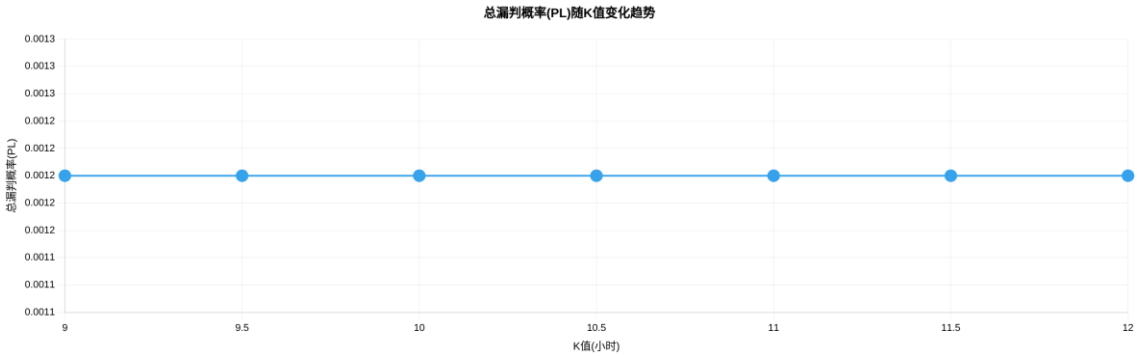
2. 通过装置数(S)随 K 值变化趋势



通过装置数(S)随 K 值变化趋势

图表显示通过装置数基本稳定在 96-100 之间，K=9.5 时出现最低值 96。

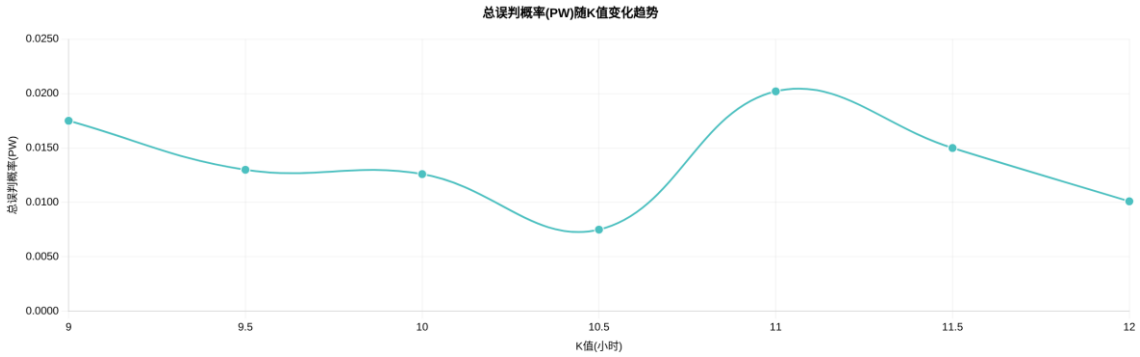
3. 总漏判概率(PL)随 K 值变化趋势



总漏判概率(PL)随 K 值变化趋势

图表显示总漏判概率在所有 K 值下保持恒定，均为 0.0012。

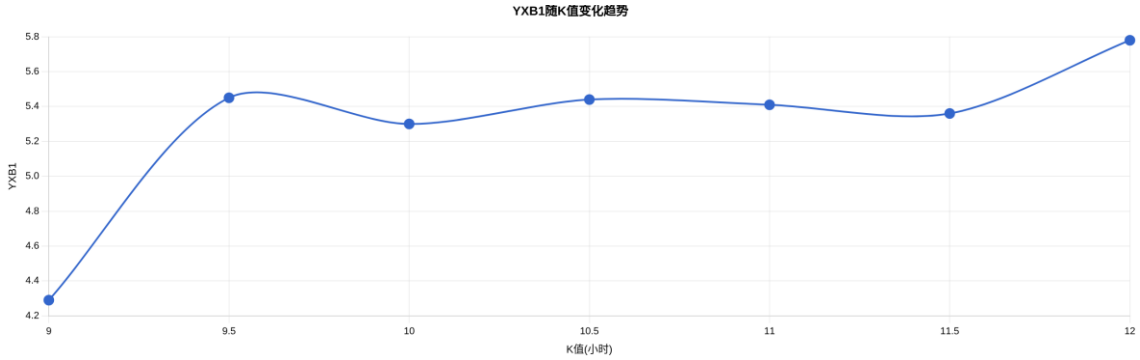
4. 总误判概率(PW)随 K 值变化趋势



总误判概率(PW)随 K 值变化趋势

图表显示总误判概率波动较大，在 K=11 时达到峰值 0.0202，在 K=10.5 时达到谷值 0.0075。

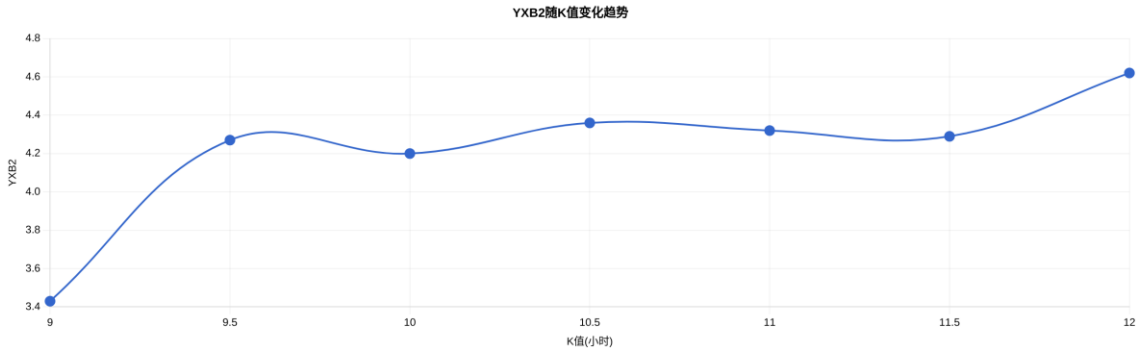
5. YXB1 随 K 值变化趋势



YXB1 随 K 值变化趋势

图表显示 YXB1 总体呈波动上升趋势，在 K=12 时达到最大值 5.78。

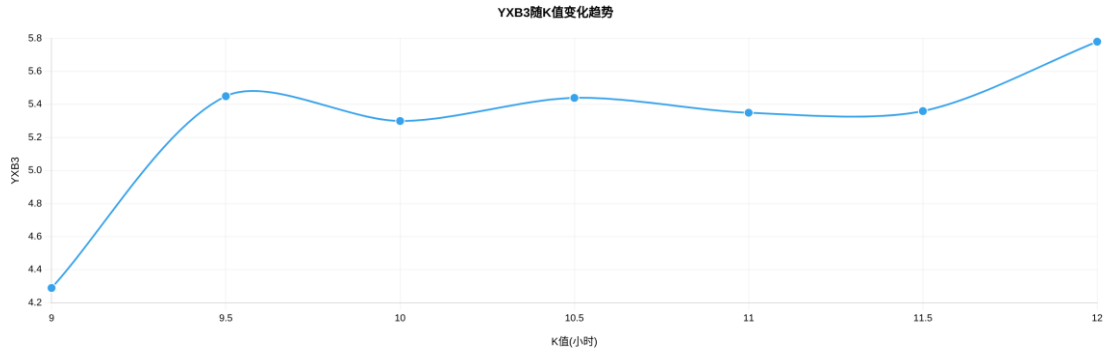
6. YXB2 随 K 值变化趋势



YXB2 随 K 值变化趋势

图表显示 YXB2 总体呈上升趋势，从 K=9 的 3.43 增长到 K=12 的 4.62。

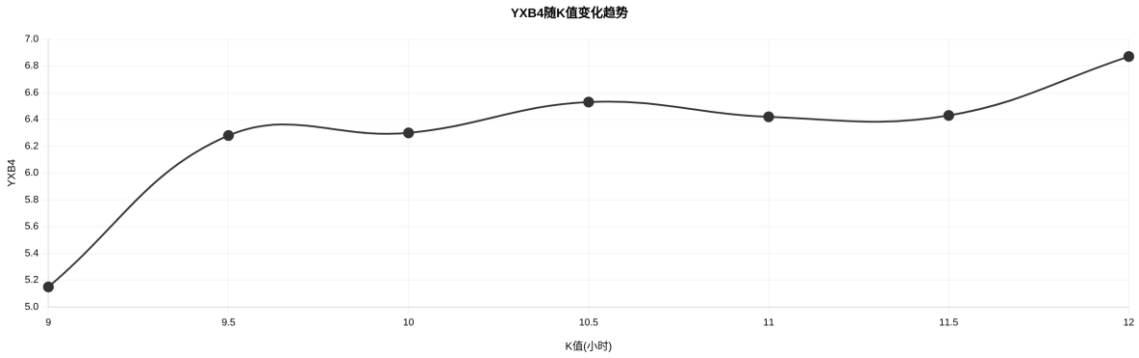
7. YXB3 随 K 值变化趋势



YXB3 随 K 值变化趋势

图表显示 YXB3 呈波动上升趋势，在 K=12 时达到最大值 5.78。

8. YXB4 随 K 值变化趋势



YXB4 随 K 值变化趋势

图表显示 YXB4 呈稳步上升趋势，从 K=9 的 5.15 增长到 K=12 的 6.87。

所有指标随 K 值变化的趋势已清晰呈现，可根据这些可视化结果分析各指标的变化规律和相关性。

附录 D 主要程序/关键代码

代码环境	操作系统: macOS Mojave (Version 10.14.3) 编程语言: Python 3.9.1 (Anaconda Navigator 1.9.2) 编辑器: PyCharm 2019.3.2 (Professional Edition) 代码详见: Code/Combine_Pyecharts_with_igraph.py

代码清单 1 NSGA-II 算法实现（测试任务调度优化）

```
import numpy as np
import matplotlib.pyplot as plt

class TestParams:
    def __init__(self):
        # 测试时间参数（单位：小时）
        self.test_time = {'A':2.5, 'B':2.0, 'C':2.5, 'E':3.0}
        self.debug_time = {'A':0.5, 'B':1/3, 'C':1/3, 'E':2/3}
        self.transport_time = 0.5  # 运入/运出时间
        self.max_shift_hour = 12   # 单班次时长
```

```
        # 随机事件概率（Y1-Y4）
        self.dev_fault_prob = {
            'A':[0.03,0.05], 'B':[0.04,0.07],
            'C':[0.02,0.06], 'E':[0.03,0.05]
        }
        self.sub_prob = {'A':0.025, 'B':0.03, 'C':0.02}
        self.operator_error = {'A':0.03, 'B':0.04, 'C':0.02, 'E':0.02}
        self.Y4_detect_prob = 0.0122  # 综合测试检出概率

        # 设备更换约束
        self.min_dev_work = 120
        self.max_dev_work = 240
        self.total_devices = 100  # 装置总数

def monte_carlo_simulation(decision, params, M=50):
    """蒙特卡洛模拟模块"""
    f1_list, f2_list = [], []
    for _ in range(M):
        dev_work_time = {'A':0, 'B':0, 'C':0, 'E':0}
        current_time = 0.0
        passed_count = 0
        total_leak = 0.0

        for device in decision['testbench1'] + decision['testbench2']:
            # 装置运入与跨班次处理
            current_time += params.transport_time
            shift_start = 24 * (current_time // 24)
            if current_time > shift_start + params.max_shift_hour:
                current_time = shift_start + 24 + params.transport_time

            # A/B/C并行测试（含设备更换与随机事件处理）
            end_time = {'A':0.0, 'B':0.0, 'C':0.0}
            for sub in ['A','B','C']:
                if (dev_work_time[sub] >= decision['dev_replace'][sub]) or \
                   (dev_work_time[sub] >= params.max_dev_work):
                    current_time += params.debug_time[sub]
                    dev_work_time[sub] = 0

                # 测试过程（支持重测）
                for retry in range(2):
                    test_duration = 0.0
                    while test_duration < params.test_time[sub]:
                        test_duration += 0.1
                        dev_work_time[sub] += 0.1
                        current_time += 0.1

                        # 随机事件检测（代码略）
                        ...

            # 综合测试E（逻辑类似ABC测试）
            ...

            # 装置运出与统计
            current_time += params.transport_time
            if test_success:
                passed_count += 1
                total_leak += params.P_leak_single

        f1_list.append(current_time)
        f2_list.append(total_leak/passed_count if passed_count>0 else 0.0)

    return np.mean(f1_list), np.mean(f2_list)

def nsga2_test_scheduling(params, N=100, G=100):
    """NSGA-II主算法"""
    # 初始化种群
    population = [encode_chromosome(params.total_devices,
                      params.min_dev_work, params.max_dev_work) for _ in range(N)]
```

```
    for gen in range(G):
        # 非支配排序与选择
        fronts, objectives = non dominated sort(population, params)
        parents = selection(population, fronts, objectives)

        # 遗传操作
        offspring = []
        for i in range(0, N, 2):
            child1 = crossover(parents[i], parents[i+1], params.total devices)
            child2 = crossover(parents[i+1], parents[i], params.total devices)
            offspring.extend([mutate(child1), mutate(child2)])

        # 种群更新
        population = update population(population+offspring, N, params)

    return get pareto front(population, params)

# 辅助函数（编码/解码/排序等）
def encode chromosome(total devices, min replace, max replace):
    """染色体编码：混合整数-实数编码"""
    # 测试台分配与测试顺序生成
    all dev = np.random.permutation(total devices)
    split idx = np.random.randint(1, total devices)
    testbench1 = all dev[:split idx].tolist()
    testbench2 = all dev[split idx:].tolist()

    # 设备更换时间（120-240h随机）
    replace times = [np.random.uniform(min replace, max replace) for    in range(4)]

    return [split_idx] + testbench1 + testbench2 + ... + replace_times  # 完整染色体

def non dominated sort(population, params):
    """快速非支配排序"""
    # 计算支配关系（代码略）
    ...
    return fronts, objectives

# 主程序
if   name    == "   main   ":
    params = TestParams()
    pareto front = nsga2 test scheduling(params)

    # Pareto前沿可视化
    plt.scatter([x[0]/24 for x in pareto front], [x[1] for x in pareto front])
    plt.xlabel('任务总时间（天）')
    plt.ylabel('总漏判概率')
    plt.show()
```

代码清单 2 融合 Pyecharts 与 igraph 模块

```
    """
    离散事件仿真程序 - 装置测试系统
    用于计算任务完成平均天数及其他性能指标
    """

    import random
    import heapq
    from dataclasses import dataclass
    from typing import List, Dict, Optional, Tuple, Union

    # ------------------------- 实体类定义 -------------------------
    @dataclass
    class Device:
```

```
    """测试装置实体类"""
    # 属性定义
    def __post_init__(self):
        # 初始化装置属性

@dataclass
class TestGroup:
    """测试小组实体类"""
    # 属性定义

    def is_fault(self) -> bool:
        # 判断设备是否故障
        pass

    def handle_error(self, device: Device) -> Tuple[bool, str, Optional[str]]:
        # 处理测手差错
        pass

@dataclass
class TestBench:
    """测试台实体类"""
    # 属性定义

@dataclass
class Event:
    """事件实体类"""
    # 属性定义

# ------------------------- 仿真核心类 -------------------------
class TestSimulation:
    """测试系统仿真核心类"""

    def __init__(self, total_devices: int = 100, shift_hours: float = 12.0):
        # 初始化仿真环境
        pass

    def _init_test_groups(self):
        # 初始化测试小组
        return {
            "A": TestGroup(...),
            "B": TestGroup(...),
            "C": TestGroup(...),
            "E": TestGroup(...)
        }

    def run(self):
        # 运行单次仿真
        while self.event_queue:
            self._process_next_event()
```

```
        return self._calculate_indices()

    def _process_next_event(self):
        # 处理下一个事件
        event = heapq.heappop(self.event_queue)
        # 事件分发处理
        pass

    def _handle_arrive(self, device):
        # 处理装置到达事件
        pass

    def _handle_assign(self, device):
        # 处理测试台分配事件
        pass

    def _handle_start_test(self, device):
        # 处理测试开始事件
        if device.current_stage == "ABC_PARALLEL":
            self._start_abc_parallel_test(device)
        elif device.current_stage == "E":
            self._start_e_test(device)

    def _start_abc_parallel_test(self, device):
        # 开始A/B/C并行测试
        pass

    def _start_e_test(self, device):
        # 开始综合测试
        pass

    def _handle_end_test(self, device):
        # 处理测试结束事件
        if device.current_stage == "ABC_PARALLEL":
            self._end_abc_parallel_test(device)
        elif device.current_stage == "E":
            self._end_e_test(device)

    def _end_abc_parallel_test(self, device):
        # 结束A/B/C并行测试
        pass

    def _end_e_test(self, device):
        # 结束综合测试
        pass

    def _handle_transport(self, device):
        # 处理运输事件
        pass
```

```
        def _calculate_indices(self):
            # 计算统计指标
            return {
                "T": total_shifts,
                "S": completed_count,
                "PL": leak_probability,
                "PW": wrong_probability,
                "YXB1": utilization_A,
                "YXB2": utilization_B,
                "YXB3": utilization_C,
                "YXB4": utilization_E
            }

    # ------------------------- 主程序 -------------------------
    if __name__ == "__main__":
        def run_multiple_simulations(num_runs=50, total_devices=100, shift_hours=12.0):
            # 运行多次仿真求平均值
            results_sum = {"T": 0.0, "S": 0.0, "PL": 0.0, "PW": 0.0,
                           "YXB1": 0.0, "YXB2": 0.0, "YXB3": 0.0, "YXB4": 0.0}

            for i in range(num_runs):
                # 单次仿真运行
                pass

            return {key: value / num_runs for key, value in results_sum.items()}

        # 运行仿真并输出结果
        avg_results = run_multiple_simulations()
        print("仿真结果:", avg_results)
```