DOI: 10.19666/j.rlfd.202505109

面向热网的多域协同冗余计算引擎设计

高德保[1]，王禹诺[2]，蔡瑞霞[3]，杨新宇[1]，王海江[1]，
魏克强[1]，邵 雷[4]，高海鹏[4]
（1. 华能聊城热电有限公司，山东 聊城 252041；
2. 西安热工研究院有限公司，陕西 西安 710054；
3. 国家能源聊城发电有限公司，山东 聊城 252000；
4. 华能能源交通产业控股有限公司北京分公司，北京 100031）

［摘 要］随着城市热网系统规模的不断扩大和智能化需求的持续增长，传统基于单控制器模式的集控系统在算力支持、系统容错性、设备兼容性以及部署成本等方面逐渐暴露出技术瓶颈，已难以满足多域协同与智能优化控制的应用需求。为此，提出一种面向热网多域协同控制的新型冗余计算引擎，采用分层架构，将计算引擎解耦为管理程序与内核程序两大模块，实现了统筹管理与核心运算功能的分离；通过任务分级管理与协调调度机制，实现了周期与非周期混合运算任务的高效协同，提高了系统的运行效率与实时响应能力；增添冗余机制，提出一种热备式冗余同步方案，提高了系统在高负载场景下的可靠性与稳定性；构建了支持多语言异构智能算法灵活接入的统一体系，通过动态加载与接口映射机制，实现了图形化组态与底层算法代码的双向联动。该计算引擎有效提高了智能算法的部署灵活性与执行效率，提高了系统运行的稳定性与可靠性，在实际部署中智能算法加载成功率 100%，主备切换时间不大于 200 ms，同时降低了传统的硬件部署成本，为工业控制系统的智能化转型提供了高性能、低成本的解决方案，并具备向能源、交通等多领域推广的潜力。

［关 键 词］热网集控系统；多域协同；工控系统智能化；热备式冗余

［引用本文格式］高德保，王禹诺，蔡瑞霞，等. 面向热网的多域协同冗余计算引擎设计[J]. 热力发电，2026，55(1)：177-186.
GAO Debao，WANG Yunuo，CAI Ruixia，et al. Design of a multi-domain collaborative redundant computing engine for the district heating network[J]. Thermal Power Generation，2026，55(1)：177-186.

Design of a multi-domain collaborative redundant computing engine for the district heating network

GAO Debao[1]，WANG Yunuo[2]，CAI Ruixia[3]，YANG Xinyu[1]，WANG Haijiang[1]，
WEI Keqiang[1]，SHAO Lei[4]，GAO Haipeng[4]
(1. Huaneng Liaocheng Thermal Power Co.，Ltd.，Liaocheng 252041，China；
2. Xi'an Thermal Power Research Institute Co.，Ltd.，Xi'an 710054，China；
3. National Energy Liaocheng Power Generation Co.，Ltd.，Liaocheng 252000，China；
4. Huaneng Energy Transportation Industry Holding Co.，Ltd.，Beijing 100031，China)

Abstract: With the continuous expansion of the scale of urban heating network systems and the sustained growth of intelligent demands，the conventional centralized control systems based on the single-controller mode have gradually exposed technical bottlenecks in terms of computing power support，system fault tolerance，equipment compatibility，and deployment costs，and have been unable to meet the application requirements of multi-domain collaboration and intelligent optimization control. To this end，a new type of redundant computing engine for multi-domain collaborative control of the heat network is proposed. A hierarchical architecture is adopted and the

收 稿 日 期：2025-05-20 修 回 日 期：2025-07-13 接 受 日 期：2025-07-16 网络首发日期：2025-10-24
基 金 项 目：中国华能集团有限公司总部科技项目（HNKJ24-H33）
Supported by：Science and Technology Project of China Huaneng Group Co.，Ltd.（HNKJ24-H33）
第一作者简介：高德保（1983），男，高级工程师，主要研究方向为暖通、热工、电气、信息、安全，gdb3234@126.com。
通信作者简介：王禹诺（2000），女，硕士研究生，主要研究方向为工控自动化系统，wangyunuo@tpri.com.cn。

computing engine is decoupled into two major modules: the management program and the kernel program, achieving the separation of overall management and core computing functions. Through the task hierarchical management and coordination scheduling mechanism, the efficient collaboration of periodic and aperiodic mixed computing tasks has been achieved, improving the operational efficiency and real-time response capability of the system. Adding a redundancy mechanism and proposing a hot standby redundancy synchronization scheme have improved the reliability and stability of the system in high-load scenarios. A unified system supporting the flexible access of multi-language heterogeneous intelligent algorithms has been constructed. Through the dynamic loading and interface mapping mechanism, the two-way interaction between graphical configuration and the underlying algorithm code has been achieved. This computing engine effectively enhances the deployment flexibility and execution efficiency of intelligent algorithms, improves the stability and reliability of system operation. In actual deployment, the loading success rate of intelligent algorithms is 100%, the primary and backup switching time is ≤200 ms, and at the same time the conventional hardware deployment cost is reduced, providing a high-performance and low-cost solution for the intelligent transformation of industrial control systems. Furthermore, it has the potential to be promoted in multiple fields such as energy and transportation.

Key words: heat-network control system; multi-domain collaboration; intelligentization of industrial control systems; hot-standby redundancy

随着城市规模的不断扩大和社会对能源利用效率要求的不断提高，热网集控系统作为城市集中供热与工业热能输送的重要基础设施，其运行规模持续增长、组成结构愈加复杂[1]，这对该系统在高效化管理与智能化升级方面提出了更高的技术要求。在当前多域协同与能效优化需求日益迫切的背景下，传统基于单个可编程逻辑控制器（programmable logic controllers，PLC）模式的热网集控系统逐渐暴露出灵活性欠缺、可扩展性差和容错性低等问题，尤其随着如模型预测控制、自适应协调优化等智能算法的涌现[2]，现有热网 PLC 控制器受限于封闭式架构，控制逻辑与专用硬件高度绑定，算法更新需增添额外的 PLC 设备，造成异构设备兼容性差、系统硬件冗杂和部署成本高等问题[3]，使得系统难以支撑智能算法的高效、安全应用，导致热网集控系统的运行效率与智能水平难以有效提升。

当前国内外在工控系统智能优化领域已开展了大量研究，尤其在模型预测控制（model predictive control，MPC）、边缘计算、智能调度优化等方面进展显著。文献[4]采用多视界模型预测控制方法，实现了分布式能源枢纽的动态调度，有效提升了能源流转效率与系统动态响应能力；文献[5]提出基于边缘计算的分布式优化框架，实现了多设备的协同计算与资源共享，推动了实时控制功能从中心服务器向边缘节点的迁移，实现了控制任务的本地化处理；文献[6]通过引入数据驱动模型和神经网络方法，提高了电热负荷预测精度并优化了碳排放效率，表明了智能算法在能效提升和碳排放控制方面的应用前景；文献[7]聚焦于 PLC 与边缘计算相融合的方案，通过算法与硬件平台的解耦，提升了系统的可扩展性与可维护性。

然而，这些方法主要聚焦于算法本身，缺乏对底层控制平台架构的优化与革新，致使传统 PLC 的封闭式架构成为热网系统智能化转型的瓶颈。对此，本文提出一种支持多域协同操作的新型冗余计算引擎，采用分层架构设计，通过模块解耦，实现了统筹管理与核心运算功能的分离；同时创新性地引入了任务分级管理与协调调度机制、热备式双冗余同步策略与智能算法的跨语言支持方案，突破了传统单 PLC 控制器在算力支持、算法扩展、系统容错和成本控制方面的瓶颈。该计算引擎不仅通过功能解耦实现了热网集控系统的高效运行，还实现了多语言异构智能算法的无损集成，为热网集控系统的智能化转型提供了核心技术支撑，具有广阔的工程应用前景与推广价值。

1 多域协同群集冗余计算引擎设计

热网系统框架如图 1 所示。其中，热网集控系统通过交换机或 VPN 网关接收各子站的运行数据，并上传至智慧供热平台实现综合监视、管理和优化控制，优化后的控制指令由集控系统下发到相应设备，最终实现对热网系统的智能化升级。

1.1 热网集控系统架构

按照自顶向下的设计原则，结合分层模块化设计，进行热网集控系统分层架构，结果如图 2 所示，这样可将热网集控系统分为用户层、管理层、内核层、硬件层。

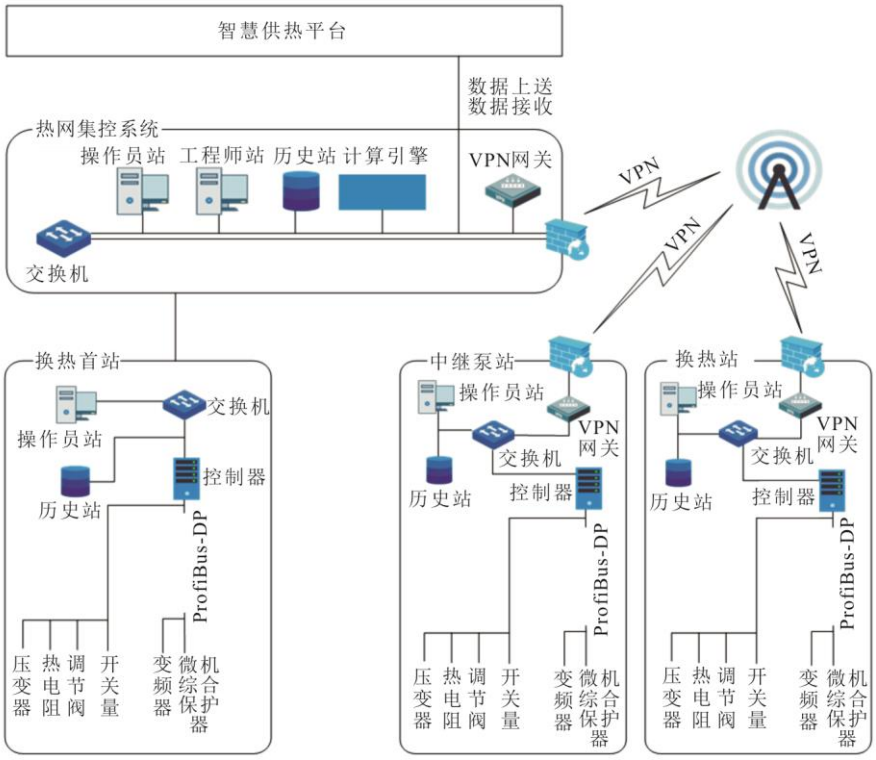


图 1　热网系统框架

Fig.1 Framework of the heat network system

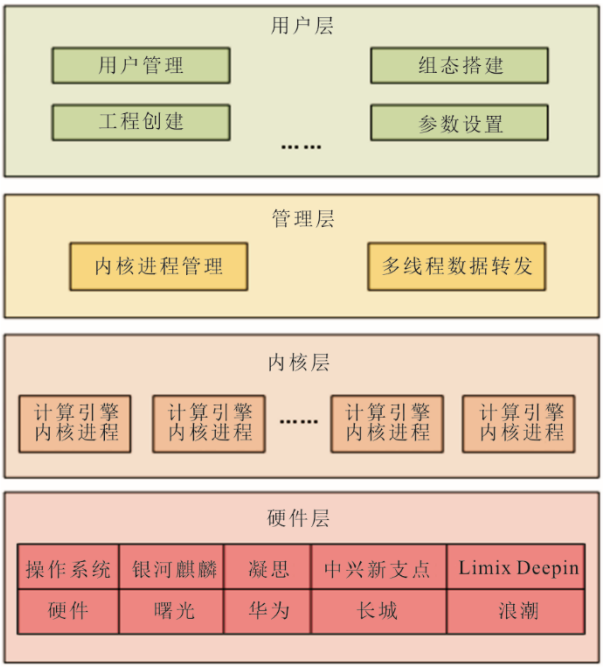


图 2　热网集控系统分层架构

Fig.2 Hierarchical architecture of the centralized control system for the district heating network

图 2 中，用户层提供基本的工程创建、组态搭建、参数设置等操作功能；管理层由内核进程管理模块和多线程数据转发模块构成，分别负责对内核软件的启停及其运行状态的监测，和内核软件与上位机、内核软件进程之间的通信；内核层用于处理控制逻辑和算法，并采用双冗余热备架构增强容错性；硬件层兼容目前主流的国产 CPU 和国产操作系统，使本计算引擎可实现跨平台灵活部署。

1.2 计算引擎软件框架

本文中的计算引擎作为可实现热网集控系统多域协同控制的核心逻辑处理单元，承担了系统的算法加载、逻辑计算、任务调度、数据通信等功能。不同于常规的数据处理框架[8]或者推理机[9]，该计算引擎更注重在工业控制场景中对混合任务的协调能力和对智能算法的承载能力。图 3 为本文设计的计算引擎软件框架。

该计算引擎包含管理软件和内核软件两部分，管理软件用于实现对内核软件的集中管理与监控，可接收上位机数据并将其下发至内核软件，同时可将内核软件的输出数据上传至上位机；内核软件是控制算法的核心运算模块，通过接收输入数据、执行算法、输出结果，实现控制功能。同时，内核软件采用双冗余热备设计，内核软件进程按主、备模式，两两分为一组，分别部署在不同服务器上，通过高效的冗余同步机制，结合故障切换策略，可保障系统的可靠性与稳定性。

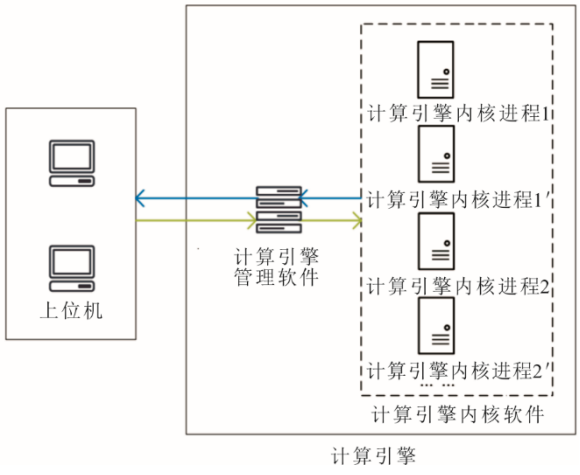


图 3 计算引擎软件框架
Fig.3 Software framework for the computing engine

1.2.1 管理软件设计

管理软件是系统上位机与内核软件的中间接口，其在代码与功能上与内核软件完全解耦，只负责对系统进行用户管理、工程管理、进程管理，并协调系统内部或是跨域的数据通信；同时具备良好的人机交互界面，可实现对内核软件数据和状态的实时监控，具体包含以下功能。

1）用户管理

构建基于角色的分层权限管理体系[10]，可创建系统管理员和普通用户 2 种角色，并配以不同权限。通过身份验证机制[11]明确用户权限，实现精细化管控，确保工业控制系统在复杂生产环境中的操作安全性与合规性。

2）工程管理

通过可视化界面展示可用 IP 地址及其关联设备在线状态，用户可直接选择或手动输入目标 IP，进行通信地址绑定与内核实例部署；对于双冗余的高可靠场景，系统强制互为主备的内核进程配置冗余 IP 地址，并内置 IP 合法性校验机制（如子网匹配、地址冲突检测），若用户未按规范配置冗余 IP 或误选非法地址（如跨网段地址、已被占用的 IP），管理软件将实时触发告警并阻断部署流程，防止因网络通信中断导致系统启动失败或控制指令丢失。

3）进程管理

可实时感知系统 CPU 核心数量、内存容量等硬件资源配置，用户通过可视化交互界面即可完成计算引擎实例的新建、删除、启停、仿真倍率调整等操作；同时采用轻量化的 UDP 网络报文与内核进程通信，实现了对系统资源的高效管理，提高了系统整体的响应速度和运行效率；通过构建多层级进程权限管控策略，从内核隔离、操作拦截、行为审计 3 个维度限制内核进程对敏感资源的访问或篡改，可有效防御越权操作、数据泄露和恶意代码注入；运行期间通过实时网络通信持续采集各内核进程的详细运行信息，在上位机中进行可视化呈现，并对异常状态进行告警，供用户及时感知内核进程的运行状态。

4）数据通信

管理软件充当内核软件与上位机、内核软件进程之间的通信数据中转站，通过多个数据转发线程，将环形缓冲区中的实时数据快速分发到相应的目标模块，其数据转发模块按功能可分为对上数据转发、对下数据转发、站间数据转发 3 类。

其中，对上数据转发指冗余内核进程将心跳信号、实时运行参数等数据通过网络报文发送给管理软件，管理软件根据不同的报文协议类型，将其转发到相应的上位机（工程师站、操作员站、历史站）中；对下数据转发指上位机在线置值或是下装组态文件时，将置值数据、组态文件等发送给管理软件，管理软件按照报文协议类型对其进行分类，再转发给对应的冗余内核进程，供内核进程修改实时运算数据或是进行组态更新；站间数据转发指当前内核进程与其他内核进程间的站间测点数据、主备冗余数据的传输，用于保证站间引用数据和主备冗余数据的一致性。

1.2.2 内核软件设计

内核软件负责系统控制逻辑的运算，支持按固定时间执行的周期运算任务和基于事件触发的非周期运算任务，同时兼容第三方智能算法。具体功能和设计方案如下。

1）组态文件数据的解析与更新

解析组态工具下装的产物文件，将其拆解为数据指令流（全局变量声明、I/O 点配置等）与逻辑指令流（常规功能块或智能算法块），再分别读取到全局变量区、硬件 I/O 区与程序组态区这 3 类独立的内存空间中，在每个区中对执行指令进行相应的赋值操作，在运算任务中可直接调度执行指令进行计算；设置动态置数区和热替换组态区双缓冲区域，分别用于接收上位机下发的实时参数和加载修改后的 POU 文件，从而实现数据的在线无扰更新。

2）混合运算任务的并行计算

建立基于执行周期与触发事件的混合运算任务模型，按照任务等级将指令流分别放入不同的处理队列中，再分别调度全局变量区、硬件 I/O 区与程序组态区进行运算。同时提出一种基于优先级的多队列顺序调度算法，结合线程池动态伸缩策略，既能保证周期运算任务的按时执行，又能有效处理非周期运算任务，避免资源冲突或任务饥饿，实现硬实时与软实时任务的高效协同。

3）内核软件的冗余同步机制

布置心跳数据点，通过定期检测心跳信号来确认主备内核进程的状态。同时设计一种热备式的冗余同步机制，包括心跳信号传递机制和冗余数据同步机制两部分，其中，心跳信号是系统的“脉搏”，决定故障检测的及时性与准确性，同步数据是系统的“血液”，保障任务状态的一致性和连续性。

1.3 本文计算引擎优势

本文计算引擎区别于单 PLC 控制器模式的热网集控系统，采用分布式运行时实例群集，通过轻量进程级虚拟化技术，实现控制逻辑执行环境与物理硬件的解耦，为热网提供弹性可扩展的智能控制平台支持；相较于 SCADA 的数据采集与上位监控导向，本方案增强了逻辑运算与算法协同功能，补全 SCADA 系统在下层控制执行能力上的不足；同时，本方案与 DCS 的层级式集中控制不同，本文计算引擎强调分布式站点独立计算与跨站直接数据交换，具备更强的多域协同与系统容错能力。表 1 为本文计算引擎与传统控制架构的对比，表 1 是在 6 个不同维度下，将本文计算引擎与传统 PLC 热网集控系统及 DCS/SCADA 间对比。

表 1 本文计算引擎与传统控制架构的对比

Tab.1 Comparison between the proposed computation engine and the conventional control architecture

维度	PLC	DCS/SCADA	本文计算引擎
架构模式	单控制器闭环	分布式 IO 站	分层解耦、双冗余运行
算法兼容性	仅支持厂商专用语言	有限脚本扩展	多语言动态加载
任务调度	固定周期扫描	固定周期扫描	混合任务分级调度
冗余机制	硬件级冗余	服务器级冗余	进程级热备
智能算法部署	新增硬件	定制开发接口	即插即用
硬件成本	高	极高	纯软件平台

2 核心技术实现

2.1 多内核进程的管理与调度

2.1.1 运算任务的分级管理

本文提出基于优先级的混合运算任务分类模型，将任务划分为 3 类，表 2 为运算任务优先级划分。其中，故障告警任务占据 0 级最高优先级，满足硬实时约束；周期运算任务占据 1—8 级优先级，执行周期分别为 5、10、20、50、100、200、500、1 000 ms，且优先级与执行周期负相关；非周期运算任务占据 9—10 级优先级，该类任务由外部事件或特定状态触发，后期支持对优先级的动态调整。

表 2 运算任务优先级划分

Tab.2 Priority division of computing tasks

优先级	类型	描述
0	故障告警	异常情况发生时的实时报警
1—8	周期运算任务	用于执行核心运算逻辑
9—10	非周期运算任务	用于执行智能算法

2.1.2 非周期任务的超时处理

非周期运算任务由于执行时间和资源需求难以预测，可能出现执行时间过长或卡死的情况，影响系统的整体稳定性。因此，需建立超时处理机制，确保非周期运算任务能及时处理异常。

创建非周期运算任务时，用户会根据其紧急性、复杂程度等因素设定一个初始超时时间。在该任务执行过程中，系统会实时监控其状态，一旦发现其执行超时，会根据如下策略进行超时延长。

1）如果 CPU 或内存的占用率较高，说明当前非周期运算任务正在进行复杂计算或受到资源瓶颈的限制，可根据式(1)对超时时间进行动态调整：

$$t_{\mathrm{ex}}=\left(\frac{\zeta_{\mathrm{cpu}}-\zeta_{\mathrm{thre}}}{100}\right)\times\delta_{\mathrm{base}} \tag{1}$$

式中：t_{ex} 为需要延长的时间；ζ_{cpu} 为当前任务占用的 CPU 资源百分比；ζ_{thre} 为正常 CPU 占用率阈值（80%）；δ_{base} 为超时延长基准（100 ms），超时以此为基础值进行延长。

2）如果非周期运算任务依赖的外部设备或数

据源在执行过程中未及时更新，系统首先根据外部资源延迟的历史数据，取其平均值作为延长时间；若外部资源仍未恢复，再设定多阶超时延长策略，分阶段动态增加超时，每阶段的延长时间根据历史延迟数据逐步增加。

3）如果给当前非周期运算任务设置的优先级较高，则可以根据式(2)给不同优先级的任务赋予不同权值，以此来动态延长超时时间。

$$t_{ex} = \eta_{pri} \times \delta_{base} \tag{2}$$

式中：η_{pri}为优先级系数，高优先级任务系数为 1.5，低优先级任务系数为 0.5。

最后，还应设置 max-t_{ex}做为最大超时延长时间（500 ms），根据式(3)保证非周期运算任务延长超时后，不会影响周期运算任务的正常执行。

$$t_{ex} = \min\left\{t_{ex}, \max\text{-}t_{ex}\right\} \tag{3}$$

如果非周期运算任务经过 3 次超时延长后仍未完成，则认为任务发生超时异常，系统分情况执行超时异常处理。

1）当非周期运算任务的 CPU 或内存占用率仍持续超过预设的阈值，导致系统资源耗尽，此时判定任务无法正常完成，终止任务；

2）当非周期运算任务依赖的外部设备仍无法恢复数据，导致数据源的响应时间超出预期，则判定任务无法执行，终止任务；

3）当非周期运算任务因临时资源问题（如内存不足、临时 IO 错误等）导致超时，鉴于该类可以通过释放资源或重新初始化环境解决，应当重启任务，重启 3 次仍未执行成功，则终止任务。

将上述对非周期运算任务的超时延长和异常处理策略总结，结果如图 4 所示。

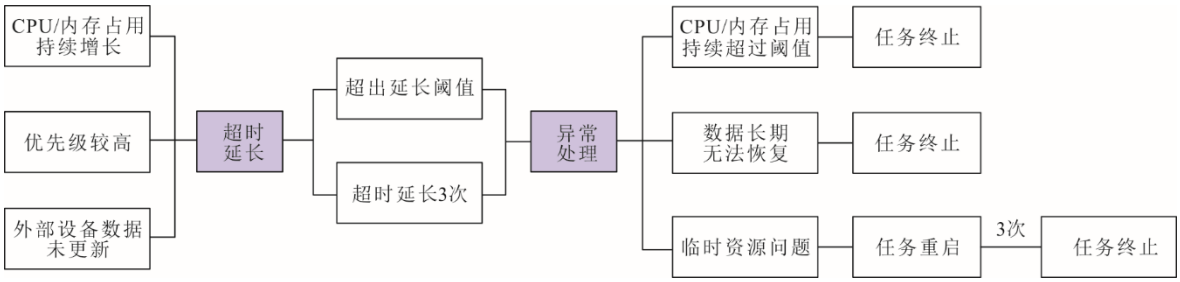


图 4 超时延长和异常处理策略

Fig.4 Timeout extension and exception handling strategies

2.1.3 混合任务的协调调度

工业生产中的运算任务调度问题可以描述为多目标最优化问题，由于实际工业生产过程中存在诸多不确定性因素，因此难以通过启发或学习式算法预先生成的调度方案获得最优解。为此，本文提出一种基于优先级的多队列顺序调度算法（图 5）。

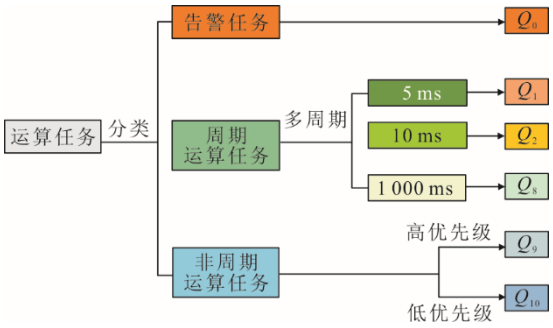


图 5 多等待队列示意

Fig.5 Multi-wait queue diagram

首先，如图 5 所示，设置 $Q_0, Q_1, \ldots, Q_{10}$共 11 个等待队列。其中，告警任务进入最高优先级的 Q_0等待队列，周期运算任务按周期长短分别进入 Q_0, Q_1, …, Q_8等待队列，非周期运算任务分别进入 Q_9, Q_{10}等待队列。

其次，每个队列内按照先来先服务（first-come, first-served，FCFS）方式进行调度，多个队列间高优先级队列先执行，且在当前等待队列为空时，才考虑执行下一级等待队列中的任务。此外，还可根据用户需要，将在 Q_9, Q_{10}中的非周期运算任务调整至 Q_1—Q_8队列，优先于部分周期运算任务执行。

通过该设计，多个运算任务按优先级分层执行，确保了高优先级周期运算任务的实时性，同时还可根据用户需要，适当提升重要非周期运算任务的优先级，以适应多样的工业生产场景。

2.2 计算引擎的双冗余热备技术

2.2.1 冗余方案设计

本文设计一种热备式冗余架构，如图 6 所示。其中，计算引擎内核软件可分为多组热备冗余的执行单元，一组热备冗余执行单元由一个主计算引擎

内核进程（主机）和一个备计算引擎内核进程（备机）构成，分别部署于独立的服务器节点，其间经计算引擎管理软件转发数据实现通信。主机负责实时执行控制任务并输出结果，备机实时同步主机的数据和状态但不参与输出；当系统监测到主机发生硬件或软件故障时，根据故障类型执行相应的同步策略，同步完成后立即触发主备切换，由备机无缝接管任务，确保控制过程的连续性和数据一致性。

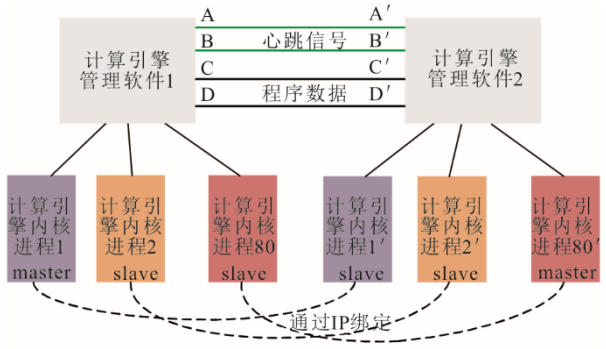


图 6 热备式冗余架构

Fig.6 Hot-standby redundant architecture diagram

2.2.2 多路通信与任务分流

本文设计了一种多路通信与任务分流策略，其中四网口双冗余架构如图 7 所示。该策略通过四网口设计，将心跳信号与数据传输分离[12]，从而实现大量数据的高效传输，降低了由网络问题导致的心跳信号丢失或是主机状态误判；同时采用双链路提升容错性，当任一网络通道发生故障时，能自动切换至备用通道继续传输。其中，心跳信号传输通道采用双路并发模式，即信号通过两条通道同时发送，一条通道的网口/链路故障时，另一通道无缝承接；数据传输通道采用双路单发模式，即数据只通过一条主通道发送，备通道作为备用仅在主通道故障时启用。通过该冗余策略，提高了系统的容错率和可靠性，优化了网络带宽的利用率。

2.2.3 冗余心跳传递

本文提出一种双向心跳推送机制，如图 8 所示。在该机制下，主备机周期性地主动向对方发送心跳报文，经由计算引擎管理软件的转发，实时监测对方的状态，并设计合理的超时检测算法，在确保高效故障检测的同时降低误判概率。

主备机周期地（默认 20 ms）发送心跳同步报文至计算引擎管理软件，管理软件监听并封装该同步报文，再转发给对机，对机解析报文后更新自身状态。图 9 为计算主备机故障切换的流程示意。

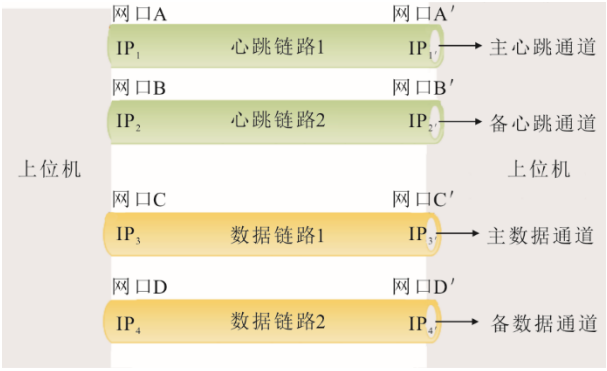


图 7 四网口双冗余架构

Fig.7 Four network ports and dual-redundancy architecture

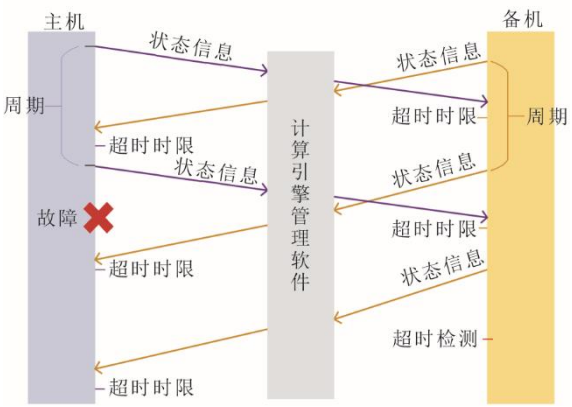


图 8 双向心跳推送机制

Fig.8 Bidirectional heartbeat push mechanism

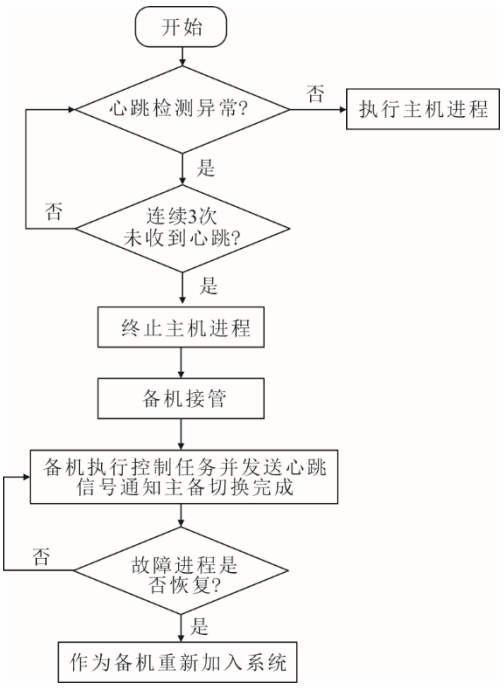


图 9 主备机故障切换流程

Fig.9 Flowchart of primary-backup failover process

若检测到主机心跳丢失，平台将自动报警并记录异常状态，立即停止主机进程，将其当前的运行数据（包括实时页面数据、IO 数据、中间变量、运算结果等）同步到备机，确保备机更新至最新状态。备机正式接管主机任务后，发送心跳信号通知如通信、报警等模块主备切换完成，所有通信与控制任务将由新主机处理，以维持平台的正常运行。当原主机恢复正常后，可以重新加入系统，作为新备机监控新主机的运行状态，此时主备机继续通过心跳机制保持同步。

2.2.4 冗余数据同步

1）周期运算任务

周期运算任务的数据同步需保证在任务执行的每个周期起始时刻，主备机的数据都保持一致，避免因数据差异导致的任务中断或运行错误。具体可分为启动前、启动时、运行时三级同步策略。

启动前同步 在系统启动或主备切换之前，通过对主备机间的非运算任务数据进行全量同步，配合 CRC 校验与数据补发机制，确保备机在启动后与主机的离线页面数据和初始运行状态保持一致。

启动时同步 在备机启动的瞬间，主机冻结全局变量与中间结果的瞬时内存快照，将快照数据与输出结果封装打包，并标记版本号和时间戳同步给备机；备机接收并检查这些数据的完整性，确保该数据能够支持备机继续执行任务后，根据最新状态接管任务。

运行时同步 当系统处于正常运行状态时，主备机之间会通过增量同步的方式，基于写时复制和脏页标记策略，仅传输发生变化的离线页面数据（POU 文件）和实时数据（如 IO 数据、置值数据等），降低带宽占用，确保主备机在任务执行过程中能够保持一致。

2）非周期运算任务

由于非周期运算任务具有触发性和偶发性，该类运算任务无需持续同步，仅确保任务触发时的数据一致即可。故采用触发式同步方式，同步包括非周期运算任务触发条件、相关联的 POU 逻辑页等任务配置信息和包括输入数据、输出结果、中间变量快照、任务元数据等上下文信息。

首先，主备机捕获到非周期运算任务触发条件，主机立即挂起当前任务线程，避免资源冲突，并通过内存快照冻结任务关联的内存区域，确保备机能够获取最新的任务状态；主机将与任务相关的配置数据和上下文信息一次性完整同步到备机，备机立即更新其内存和任务状态，确保主备机的数据一致性，同时备机会冻结与当前任务相关的内存区域，为故障切换接管主机任务做准备；数据同步完成后，非周期运算任务仅在主机上执行，备机不参与和主机的实时数据同步；非周期运算任务结束后，备机也不进行实时数据同步，直到下一次任务触发。图 10 为非周期运算任务的触发式数据同步示意。

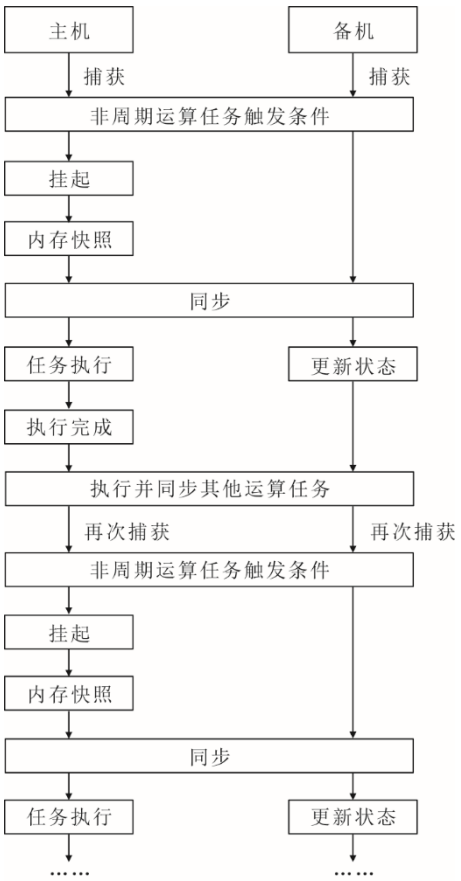


图 10 触发式同步示意

Fig.10 Diagram of triggered synchronization

2.3 计算引擎对智能算法的支持

2.3.1 模板设计

为高效集成基于多语言编写的异构智能算法，本文设计了一种标准化算法模板，具体包括：

1）统一数据结构体

主要包括描述智能算法变量名称、数据类型、数据类别（输入、输出、参数、中间变量）以及变量值的结构体，以及用于描述智能算法运行结果（执行成功、输入无效、溢出错误等）的结构体等。

2）核心函数接口

主要包括核心执行函数 fb_run、变量存取函数

updateVariable 等、变量读写函数 getIntValue、setIntValue 等。

3）元数据管理

主要包括元数据获取函数 getPouInfo，用于获取功能块的基本信息，返回一个包含功能块名称、版本以及输入输出变量列表等元数据的结构体。

2.3.2 封装与编译

针对不同编程语言的特性，构建编译转换工具链，将多语言智能算法封装为动态链接库，从而实现其在系统中的注册、装载、调度与释放。

通过 GCC 编译器直接将 C、C++代码封装为动态链接库；或是通过 Cython 将 Python 脚本封装为 C 语言动态库，通过 MATLAB Coder 转换 Simulink 模型为 C 语言代码，再结合 GCC 编译器生成与系统适配的动态链接库。通过该工具链，可实现多语言算法的无缝集成，并具备运行时动态加载与释放的能力。

2.3.3 代码−图形双向映射

构建以数据描述文件（.dblx）和图形配置文件（.gblx）为核心的双层 XML 文件体系[13-14]。其中，数据描述文件负责声明功能块元数据，如算法名称、输入输出变量、接口函数等；图形配置文件包含智能算法的图形化功能块表示信息，如节点位置、端口布局、连线关系等。系统通过读取 XML 文件，将控制逻辑代码动态渲染成图形化的智能算法功能块，支持用户进行拖拽、连接等操作形成新控制流程，并可反向驱动对应代码的生成与更新。图 11 为智能算法功能块的代码−图形双向更新流程。

用户可直接编辑智能算法的源代码，即修改 fb_run 函数的核心逻辑，系统会自动解析接口函数的代码变化，同步更新图形化界面；同时，用户在图形化界面修改功能块的参数或连接关系时，会直接修改代码的全局变量，或重构数据流代码并重新生成调度逻辑。

2.3.4 解析与调用

通过宏定义区分 Windows/Linux，封装环境差异，保证对不同系统的兼容性。在 Windows 平台下，使用 LoadLibrary 函数加载动态库文件（.dll），通过 GetProcAddress 函数获取动态库导出函数接口；在 Linux 平台下，使用 dlopen 函数加载动态库文件（.so），通过 dlsym 函数获取动态库导出函数接口。

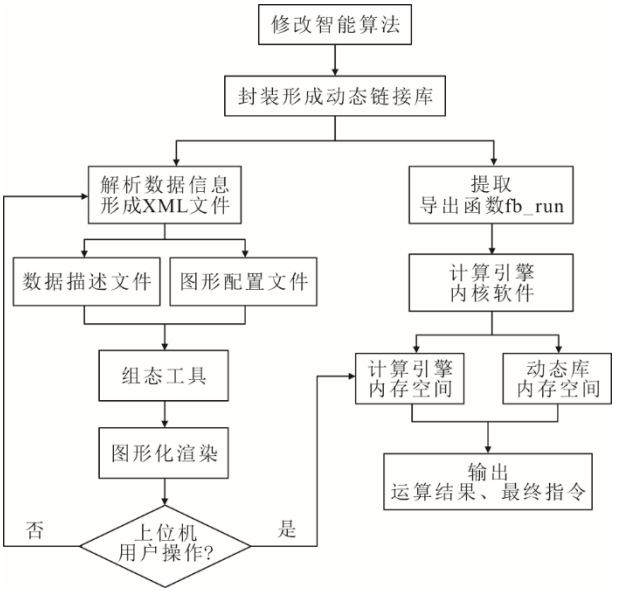


图 11 代码−图形双向更新流程

Fig.11 Code-graphic bidirectional update flowchart

通过添加关键字段对智能算法功能块与原生功能块加以区分，并使用 map 机制建立功能块名称与动态库核心接口函数 fb_run 的映射关系，实现系统对智能算法的集成。同时开辟 2 个独立的内存空间，支持运行时更新智能算法的输入、参数变量，通过执行动态库中的导出函数、返回运算结果，实现对智能算法的动态调用。

3 结　语

在热网集控系统智能化转型的需求下，本文提出并实现了一种多域协同群集新型冗余计算引擎，通过分层解耦的架构设计，成功解决了传统 PLC 控制器在算法兼容性、算力支撑及冗余成本方面的瓶颈。该计算引擎将统筹管理与逻辑运算功能分离，结合任务分级与混合调度策略，实现了周期与非周期运算任务的高效协同；同时创新性地提出了多路通信与任务分流策略、双向心跳推送机制以及非周期运算任务的触发式同步方案，将主备计算引擎的切换时间稳定在 200 ms 以内，不仅满足了热网系统在实时性与稳定性方面的需求，还大幅降低了网络资源争用带来的潜在风险；最后通过构建标准化算法模板与跨语言编译工具链，实现了异构智能算法的灵活接入与动态加载，经连续 300 次试验，Python/MATLAB/C++算法的载入成功率为 100%，为热网控制系统的智能化升级提供了核心技术支撑。

随着智能算法复杂度的持续提升与工业场景的多元化需求，本文开发的计算引擎仍有优化空

间。首先，可探索基于边缘计算的分布式架构，结合轻量化容器技术，实现资源弹性分配与动态扩展，以应对超大规模热网系统对实时性的高要求；其次，还需深化与人工智能技术的融合，引入自学习与自适应机制，使计算引擎能够根据历史数据与实时工况动态调整调度策略，提升系统的自主决策能力；此外，跨领域协同控制的通用性扩展也是重要的优化方向，通过构建统一协议适配框架与标准化接口，可以推动该计算引擎在能源、交通等多领域的应用；最后，工业控制系统面临的安全风险日趋严峻，未来研究应强化安全防护机制，结合可信计算、动态加密等技术，确保系统在复杂网络环境下的数据完整性与运行可靠性，为工业控制系统的全面数智化转型提供坚实可靠的安全支撑。

[参 考 文 献]

[1] 钟绍松, 杨锦松, 许世龙, 等. 智能化热网运行管控系统建设与应用[J]. 网络安全技术与应用, 2020(10): 151-153.
ZHONG Shaosong, YANG Jinsong, XU Shilong, et al. Construction and application of intelligent heat network operation and control system[J]. Network Security Technology and Application, 2020(10): 151-153.

[2] 国海龙. 城市智能热网综合管控系统应用研究[D]. 北京: 华北电力大学, 2014: 1.
GUO Hailong. Application research of city intelligent heating network integrated management control system[D]. Beijing: North China Electric Power University, 2014: 1.

[3] 赵琰. 软件定义工业控制器关键技术研究与实现[D]. 广州: 广州大学, 2022: 1.
ZHAO Yan. Research and implementation of key technologies for software-defined industrial controllers[D]. Guangzhou: Guangzhou University, 2022: 1.

[4] BEHRUNANI V, CAI H, HEER P, et al. Distributed multi-horizon model predictive control for network of energy hubs[J]. Control Engineering Practice, 2024, 147: 2304.

[5] 王雪. 基于分布式优化的移动边缘计算任务调度和设备协同算法研究[D]. 北京: 北京交通大学, 2019: 1.
WANG Xue. Research on task scheduling and device collaboration algorithm for mobile edge computing based on distributed optimization[D]. Beijing: Beijing Jiaotong University, 2019: 1.

[6] 潘雨情. 园区电热负荷可调节潜力分析及需求响应策略研究[D]. 北京: 华北电力大学, 2024: 1.
PAN Yuqing. Analysis of the adjustable potential of electric heating load in the park and research on demand response strategies[D]. Beijing: North China Electric Power University, 2024: 1.

[7] 邢汉, 熊光亚, 刘林兴, 等. 一种分布式 PLC 边缘计算系统及数据通信方法[J]. 水电与抽水蓄能, 2024, 10(1): 78-83.
XING Han, XIONG Guangya, LIU Linxing, et al. A distributed PLC edge computing system and data communication method[J]. Hydropower and Pumped Storage, 2024, 10(1): 78-83.

[8] DEAN J, GHEMAWAT S. Mapreduce: simplified data processing on large clusters[J]. OSDI, 2004, 4(1): 5.

[9] STUDER R, BENJAMINS V R, FENSEL D. Knowledge engineering: principles and methods[J]. Data & Knowledge Engineering, 1998, 25(1/2): 161-197.

[10] 马焕鹏, 叶荣华. SaaS 系统访问控制模型的研究[J]. 电脑与电信, 2016(增刊 1): 78-80.
MA Huanpeng, YE Ronghua. Research of access control model for SaaS system[J]. Computer and Telecommunication, 2016(Suppl.1): 78-80.

[11] 刘新, 孙强, 于灏, 等. 电力信息系统安全关键技术分析与研究[J]. 自动化技术与应用, 2019, 38(4): 63-68.
LIU Xin, SUN Qiang, YU Hao, et al. Analysis and research on key security techniques in electric power information system[J]. Techniques of Automation and Application, 2019, 38(4): 63-68.

[12] 王禹诺, 高少华, 曹桦松, 等. 基于软件定义的新型协同控制平台设计[J]. 热力发电, 2024, 53(12): 143-150.
WANG Yunuo, GAO Shaohua, CAO Huasong, et al. Design of a new collaborative control platform based on software definition[J]. Thermal Power Generation, 2024, 53(12): 143-150.

[13] 朱灿鹏. 基于 XML 文档的监控组态软件设计[D]. 南京: 南京理工大学, 2014: 1.
ZHU Canpeng. Design of monitoring configuration software based on XML documents[D]. Nanjing: Nanjing University of Science and Technology, 2014: 1.

[14] 刘瑞祥, 柳先辉, 赵卫东, 等. 基于大语言模型的业务流程自动建模方法[J]. 计算机集成制造系统, 2025, 31(6): 2001-2014.
LIU Ruixiang, LIU Xianhui, ZHAO Weidong, et al. Automatic business process modeling method based on large language models[J]. Computer Integrated Manufacturing System, 2025, 31(6): 2001-2014.

（责任编辑　杜亚勤）

广告目次

《热力发电》第五届编委会……封三
浙江顺豪新材料有限公司……后彩插 1
《热力发电》订阅……后彩插 2