DOI: 10.7641/CTA.2017.60350

并行卡尔曼滤波器的多信道序列规划算法

陆诗远, 颜钢锋†, 林　澈, 郑荣濠

(浙江大学 电气工程学院, 浙江 杭州 310058)

摘要: 针对基于卡尔曼滤波器的并行序列规划问题, 本文提出了一种基于多信道的改进算法, 在规划各进程所对应传感器的测量信息传输序列, 以使相应滤波器的状态估计误差协方差满足给定指标的同时, 尽可能减少通讯信道占用. 文中论证了基于夹挤方法的周期算法, 以提高计算进程传输周期的效率; 改进了基于缓存的离线序列规划算法, 以适应多信道传输的情景; 设计了在线序列调整算法, 利用空闲信道进行传输失败后的数据复发, 快速调整序列以适应指标变化的需求. 算法的运行效果将通过仿真实验进行验证与分析.

关键词: 卡尔曼滤波; 并行估计; 传感器规划; 多信道

中图分类号: TP273　　**文献标识码**: A

Improved multi-channel scheduling algorithm for parallel Kalman filters

LU Shi-yuan, YAN Gang-feng†, LIN Che, ZHENG Rong-hao

(College of Electrical Engineering, Zhejiang University, Hangzhou Zhejiang 310058, China)

Abstract: In this paper, an improved multi-channel algorithm is designed to schedule parallel processes of Kalman filters. The goal is to schedule the transmission of sensor measurements for all processes to meet individual estimation error requirements with the least communication channels occupied. Our algorithm includes the following methods: A fast cycle estimation method based on the squeeze method, which accelerates the search for transmission cycles significantly; An off-line schedule method based on the Buffer Scheme, which is suitable for scheduling transmissions with multiple communication channels; An on-line reschedule method, which utilizes idle channels against transmission losses and adjusts the transmission sequence quickly in case of process requirement changes. The effect of the algorithm is verified by simulations.

Key words: Kalman filter; parallel estimation; sensor scheduling; multi-channel

1　引言(Introduction)

材料科学和制造工艺的进步促使传感器趋于智能化、小型化和廉价化, 导致无线传感器网络的大规模应用. 在无线网络中, 传感器具备感应、运算和通讯等功能, 依赖电池等储能设备供能. 在有限的能量下, 如何选择传感器, 合理安排运行时间和方式, 以优化任务目标[1], 一直以来都是传感器序列规划问题所关注的.

卡尔曼滤波器具备线性最优的特性, 被广泛应用于传感器网络中. 基于卡尔曼滤波器的序列规划, 是近年来研究的热点. Gupta等[2]提出, 可通过切换不同传感器的观测传输序列, 优化单一进程的估计误差协方差, 是最早的相关研究之一. 针对同一问题, Sandberg等[3]将两种不同精度和延迟的传感器作为研究对象, 并发现最优的传输序列具备周期性. Mo等[4]证明若序列导致的平均协方差有界, 必存在周期序列达成近似效果. Orihuela等[5]总结, 在一定条件下, 基于卡尔曼滤波器的规划, 总是存在形为周期序列的解. 上述问题中, 传感器与进程呈多对一关系, 当这种关系呈现一一对应时, 序列规划以决定传感器是否(传输)观测的方式实现. Sinopolic等[6]探讨了当观测序列服从二项分布时, 观测概率范围与期望误差协方差收敛性的关系; Rohr等[7]推导了更精确的概率上下界, 并进一步讨论了误差协方差的分布. 针对降低平均传输误差的随机传输问题, 肖力等[8]讨论了传感器切换和传输错漏的均服从二项分布时, 最优切换概率的计算; 刘永桂等[9]设计了适应错漏和时延的最优算法, 并将其拓展到时变和非线性系统. 为获得非随

收稿日期: 2016−05−26; 录用日期: 2016−09−28.

†通信作者. E-mail: ygf@zju.edu.cn.

本文责任编委: 俞立.

浙江省自然科学基金(R13F030002)资助.

Supported by Zhejiang Provincial Natural Science Foundation of China (R13F030002).

机的最优序列, Ren等[10]设计了基于马尔可夫链的寻优算法, 以有限的传输获得最优的平均误差. Li等[11]认为不同的能量传输的误码概率不同, 提出了降低周期平均误差的能量最优分配方法. Shi等[12]论证了平均协方差最优的离线序列中观测间隔是相等的, 并提出了基于事件触发(event-trigger)机制的在线序列规划方法. 前述研究注重单一进程的排序优化, 而实际传感网络中难免出现多进程并行的情况, 为避免传输信道冲突, 并使各进程的误差协方差均满足给定指标, Lin等[13]利用序列的周期性设计了并行规划算法. Lu等[14]基于滤波器的间断观测特性, 提出基于周期时限计算的动态调整算法, 使序列规划能适应传输错漏和指标变化.

针对文献[14]的下述不足, 本文设计了改进算法: 原文周期计算所依赖的协方差上界, 存在对系统矩阵的特殊限制, 且不足以获得精确的传感器传输周期; 基于单信道的序列规划算法, 难以适应大量进程存在需要更多信道通讯的现实. 本文工作可概述为: 设计了基于夹挤方法的快速周期算法, 适用范围广、周期计算准确. 综合Windows Scheduling的研究成果, 设计基于缓存的多信道离线规划算法. 在线序列调整部分, 改进了传输错误后利用空闲信道复发的规则, 设计契合缓存算法的指标变化后的重规划流程, 算法更符合无线网络的现实条件. 本文算法的改进效果将通过仿真实验进行验证和分析.

定义 1 对于矩阵A和B, $A < B(A \leqslant B)$表示矩阵$A-B$(半)负定. "⌈ ⌉"是向上取整符号, $\lceil 1.2 \rceil = 2$.

2 问题描述(Problem formulation)

本文研究的系统由n个独立进程组成(见图1), 任意进程s_i的状态x^i由相应的传感器i检测, 其观测输出为y_i. 控制中枢通过无线传感网络接收观测信息, 输入对应的滤波器, 估计各进程状态. 为避免通讯干扰, 同一时刻内来自不同传感器的观测信息占用的信道不同; 因网络传输限制, 同一时刻系统中最多有m个滤波器能获得观测信息.

任意进程s_i的动态过程可表征如下:

$$\begin{aligned} x_k^i &= A^i x_{k-1}^i + w_k^i, \\ y_k^i &= C^i x_k^i + v_k^i, \end{aligned} \tag{1}$$

其中: $x_k^i \in \mathbb{R}^{p^i}$和$y_k^i \in \mathbb{R}^{q^i}$为$k$时刻$s_i$的状态和测量值; $w_k^i \in \mathbb{R}^{p^i}$和$v_k^i \in \mathbb{R}^{q^i}$为过程噪声和测量噪声, 是相互独立且分别服从$\mathcal{N}(0, Q^i)$和$\mathcal{N}(0, R^i)$分布的白噪声; A^i和C^i为非时变的状态转移矩阵和观测矩阵. 此外, 上述矩阵满足(A^i, C^i)能观、$(A^i, \sqrt{Q^i})$能控.

进程在控制中枢的状态估计值的获取, 通过运行相应的具间断观测功能的卡尔曼滤波器实现, 在k时刻对应于进程s_i的滤波器可由下述方程描述:

$$\begin{cases} \bar{x}_k^i = A^i \hat{x}_{k-1}^i, \\ \bar{P}_k^i = A^i P_{k-1}^i (A^i)^{\mathrm{T}} + Q^i, \\ K_k^i = \bar{P}_k^i (C^i)^{\mathrm{T}} [C^i \bar{P}_k^i (C^i)^{\mathrm{T}} + R^i]^{-1}, \\ \hat{x}_k^i = \bar{x}_k^i + \lambda_k^i K_k^i (y_k^i - C^i \bar{x}_k^i), \\ P_k^i = (I - \lambda_k^i K_k^i C^i) \bar{P}_k^i, \end{cases} \tag{2}$$

其中: $\bar{x}_k^i$为基于系统参数和历史信息的先验状态估计, $\hat{x}_k^i$为基于当前测量信息的后验状态估计, $\bar{P}_k^i$和P_k^i是相应的估计误差协方差; λ_k^i是一个二值的常量, 当$\lambda_k^i = 0$时, 滤波器(控制中枢)未收到s_i的测量信息, 根据先验信息获得对系统的估计; 当$\lambda_k^i = 1$时, 上述是一个标准的卡尔曼滤波器, 将在先验估计基础上进行后验估计.

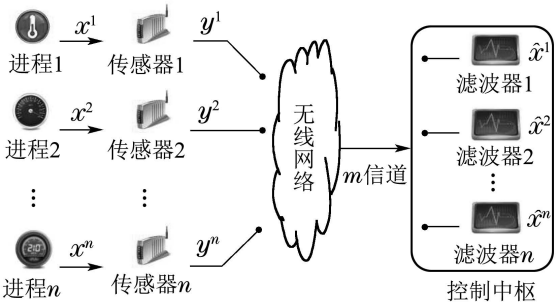

图 1 系统示意图

Fig. 1 System framework

注意到方程(2)中估计误差协方差的变化与测量值y_k^i无关, 与是否收到测量信息, 即λ_k^i有关. 因此, 任意k时刻进程s_i的误差协方差P_k^i是由其初始条件P_0^i和观测信息的传输序列$\lambda^i = \{\lambda_1^i, \lambda_2^i, \cdots\}$共同决定的. 令$\lambda = \{\lambda^1 \cup \lambda^2 \cup \cdots \cup \lambda^n\}$表征所有传输序列的集合, 根据系统设定, 单位时刻内通讯受限, 则序列λ中k时刻所有λ_k^i须满足

$$\sum_{i=1}^{n} \lambda_k^i \leqslant m. \tag{3}$$

本文研究的问题可概述为: 给定进程的误差协方差指标$P_{\sup}^i (i = 1, 2, \cdots, n)$, 规划序列$\lambda$, 使任意$k$时刻任意进程$s_i$的误差协方差$P_k^i$均满足$P_k^i - P_{\sup}^i \leqslant 0$, 并尽可能减少传输占用的通信信道数量$m$.

3 算法实现(Scheduling algorithm)

本文设计的算法由3部分组成, 按执行顺序依次为: 传输周期计算, 计算各进程满足指标所允许的最大传输周期, 以作为后续规划依据; 离线序列规划, 通过安排协调传感器的传输顺序, 使各进程的实际周期不大于前述最大值, 并减少占用的信道数量; 在线动态调整, 针对实际运行中的传输错漏或指标变更, 调整前一部分所规划的序列, 降低扰动影响.

3.1 传输周期计算(Transmission cycle calculation)

本节中描述基于夹挤的快速周期计算方法的原理和实现. 为简化后续论述, 定义

$$\begin{cases} h_i(X) \triangleq A^iX(A^i)^{\mathrm{T}}+Q^i, \\ g_i(X) \triangleq X-X(C^i)^{\mathrm{T}}[C^iX(C^i)^{\mathrm{T}}+R^i]^{-1}C^iX, \\ h_i^j(X) \triangleq \underbrace{h_i(h_i(\cdots h_i}_{j个}(X))), \\ f_i(X,j) \triangleq g_i(h_i^j(X)), \end{cases} \tag{4}$$

则k时刻式(2)中进程s_i的协方差P_k^i的单步更新方程可表述为

$$P_k^i=\begin{cases} h_i(P_{k-1}^i), & \lambda_k^i=0, \\ f_i(P_{k-1}^i,1), & \lambda_k^i=1. \end{cases} \tag{5}$$

根据文献[15]附录中的论述, 下述引理成立:

引理 1　若矩阵X和Y满足$0\leqslant X\leqslant Y$, 则

$$\begin{cases} h_i(X)\leqslant h_i(Y), \\ g_i(X)\leqslant g_i(Y), \\ g_i(X)\leqslant X. \end{cases} \tag{6}$$

由引理1可知, 式(5)中$f_i(P_{k-1}^i,1)\leqslant h_i(P_{k-1}^i)$, 因此结合测量信息的后验估计的误差协方差总是不劣于仅依据模型的先验估计. 然而测量信息传输越多, 信道占用率和能量消耗也越高, 计算满足各进程指标的最大传输间隔正是我们所需要的.

先回顾下文献[13]中对上述间断观测的卡尔曼滤波器的研究, 据其章节3.2的分析, 下述引理成立:

引理 2　当(A^i,C^i)能观、$(A^i,\sqrt{Q^i})$能控时, 若卡尔曼滤波器的观测传输序列以N为周期, 即$\lambda_{tN}^i=1,\lambda_{tN+1}^i=\lambda_{tN+2}^i=\cdots=\lambda_{tN+N-1}^i=0, t\in\mathbb{N}, t\to\infty$时其估计误差协方差必将收敛于下述矩阵

$$\begin{aligned} &\lim_{t\to\infty}P_{tN}^i=\check{P}_N^i, \\ &\lim_{t\to\infty}P_{tN+j}^i=h_i^j(\check{P}_N^i),\ j=1,2,\cdots,N-1, \end{aligned} \tag{7}$$

其中$\check{P}_N^i$是方程$\check{P}_N^i=f_i(\check{P}_N^i,N)$的唯一解, 是一个半正定矩阵.

结合引理2, 文献[13]论证下述假设1成立时, 对于任意满足上述能观性和能控性条件的滤波器, 总能找到一个最大的间断观测周期$N_{\max}^i$, 使$P_k^i\leqslant h_i^{N_{\max}^i-1}(\check{P}_{N_{\max}^i}^i)\leqslant P_{\sup}^i$, 但并未提出有效的周期算法.

假设 1　令$\check{P}_N^i$为观测周期为N的滤波器后验估计后的稳态协方差, 对任意满足$j\leqslant k$的自然数, 均有$h_i^j(\check{P}_N^i)\leqslant h_i^k(\check{P}_N^i)$. 即仅进行基于模型的先验估计, 将导致估计误差协方差持续增大.

基于该假设, 本文提出了下述定理, 作为本节周期计算方法的依据:

定理 1　当进程s_i的观测序列分别以N_j和N_k为周期时, 若$N_j\leqslant N_k$, 则有: a) $\check{P}_{N_j}^i\leqslant\check{P}_{N_k}^i$; b) $h_i^{N_j-1}(\check{P}_{N_j}^i)\leqslant h_i^{N_k-1}(\check{P}_{N_k}^i)$.

证　a) 设进程s_i以周期N_j和N_k的两种方式分别进行更新, 其协方差初始条件记为$P_j(0)=P_k(0)=\check{P}_{N_j}^i$. 第2个周期分别开始时, 即$N_j$时刻$P_j(N_j)=f_i(\check{P}_{N_j}^i,N_j)=\check{P}_{N_j}^i$; N_k时刻$P_k(N_k)=f_i(\check{P}_{N_j}^i,N_k)$, 根据假设1, $h_i^{N_j}(\check{P}_{N_j}^i)\leqslant h_i^{N_k}(\check{P}_{N_j}^i)$, 则$\check{P}_{N_j}^i\leqslant P_k(N_k)$. 在第3个周期时, $P_j(2N_j)=\check{P}_{N_j}^i=f_i(\check{P}_{N_j}^i,N_j)\leqslant f_i(P_k(N_k),N_j)=P_k(2N_k)$. 以此类推, 可得$P_j(tN_j)=\check{P}_{N_j}^i\leqslant P_k(tN_k)$. 根据引理2, $\lim\limits_{t\to\infty}P_k(tN_k)=\check{P}_{N_k}^i$, 因此$\check{P}_{N_j}^i\leqslant\check{P}_{N_k}^i$.

b) 结合定理1的a)和假设1可以得到$h_i^{N_j-1}(\check{P}_{N_j}^i)\leqslant h_i^{N_j-1}(\check{P}_{N_k}^i)\leqslant h_i^{N_k-1}(\check{P}_{N_k}^i)$.　　证毕.

注 1　根据引理1可证$\check{P}_N^i=f_i(\check{P}_N^i,N)\leqslant h_i^N(\check{P}_N^i)$, 却不足以进一步证明假设1始终成立. 在本文实现中, 本文采取假设其成立, 并于计算周期后验证的方法. 本文实验中的案例均满足假设, 未来的工作将包括对该假设的证明.

基于定理1, 本文设计了一种夹挤算法, 以避免遍历计算最大周期$N_{\max}^i$, 其流程描述如下:

算法 1　最大传输周期计算.

输入: 参数A^i,C^i,Q^i,R^i, 指标$P_{\sup}^i$.

输出: 最大周期$N_{\max}^i$.

1) 初始化, 步序$k=0$, $N_{\mathrm{lb}}^i(0)=1$, $N_{\mathrm{ub}}^i(0)=\infty$.

2) $k=k+1$, 计算$\check{P}_{N_{\mathrm{lb}}^i(k)}^i=f_i(\check{P}_{N_{\mathrm{lb}}^i(k)}^i,N_{\mathrm{lb}}^i(k-1))$.

3) $N_{\mathrm{ub}}^i(k)=\max\{t\in\mathbb{N}:h_i^t(\check{P}_{N_{\mathrm{lb}}^i(k)}^i)\leqslant P_{\sup}^i\}+1$. 若$N_{\mathrm{ub}}^i(k)=N_{\mathrm{lb}}^i(k-1)$, 转流程7); 若$N_{\mathrm{ub}}^i(k)=N_{\mathrm{ub}}^i(k-1)$, 转流程8).

4) 计算$\check{P}_{N_{\mathrm{ub}}^i(k)}^i=f_i(\check{P}_{N_{\mathrm{ub}}^i(k)}^i,N_{\mathrm{ub}}^i(k))$.

5) $N_{\mathrm{lb}}^i(k)=\max\{t\in\mathbb{N}:h_i^t(\check{P}_{N_{\mathrm{ub}}^i(k)}^i)\leqslant P_{\sup}^i\}+1$. 若$N_{\mathrm{lb}}^i(k)=N_{\mathrm{ub}}^i(k)$, 转流程7); 若$N_{\mathrm{lb}}^i(k)=N_{\mathrm{lb}}^i(k-1)$, 转流程8).

6) 重复流程2)–5).

7) 输出结果$N_{\max}^i=N_{\mathrm{ub}}^i(k)$.

8) 遍历区间$[N_{\mathrm{lb}}^i(k),\ N_{\mathrm{ub}}^i(k)]$所有整数, 计算$N_{\max}^i=\max\{t\in[N_{\mathrm{lb}}^i(k),N_{\mathrm{ub}}^i(k)]:h_i^t(\check{P}_t^i)\leqslant P_{\sup}^i\}+1$, 输出结果.

上述算法中, $N_{\mathrm{lb}}^i(k)$和$N_{\mathrm{ub}}^i(k)$是第k步迭代时计算的周期$N_{\max}^i$上、下界. 迭代过程中$\check{P}_{N_{\mathrm{lb}}^i(k-1)}^i\leqslant\check{P}_{N_{\mathrm{lb}}^i(k)}^i\leqslant P_{\sup}^i\leqslant\check{P}_{N_{\mathrm{ub}}^i(k)}^i\leqslant\check{P}_{N_{\mathrm{ub}}^i(k-1)}^i$, $N_{\mathrm{lb}}^i(k-1)\leqslant N_{\mathrm{lb}}^i(k)\leqslant N_{\max}^i\leqslant N_{\mathrm{ub}}^i(k)\leqslant N_{\mathrm{ub}}^i(k-1)$(根据定理1). 算法在流程2)和5)之间迭代循环. 流程8)的条件成立

时, 无法继续迭代. 根据获得的$N^i_{\max}$所在的区间, 通过遍历方式计算.

3.2 离线序列规划(Off-line scheduling algorithm)

本节将基于最大周期规划的传感器传输序列. 这是一个周期固定的重复序列, 忽略了随机传输错误和指标变化, 实质上是离线的. 算法由以下部分组成: 周期规范化, 生成规范长度的传输周期, 以利于之后的序列规划; 空进程引进, 允许信道空闲, 使进程周期严格服从规范后的长度; 缓存序列规划, 逐步生成传输序列. 本文中控制中枢具有所有进程的参数信息, 因此可在初始化时以本节算法完成所有时刻的离线规划, 并告知各传感器.

本节中的算法是基于Windows Scheduling问题的研究. 此类问题始于文献[15]中的研究, 结合本文应用, 可将其概述为: 给定进程$s_1, s_2, \cdots, s_n$及其各自对应的整数周期$N^i_{\max} \geqslant 1$, 将其信息传输安排在m信道中, 并设同一时刻每个信道仅对应于一个进程, 寻找占用信道最少的传输序列并满足所有进程的传输间隔均不大于$N^i_{\max}$. 文献[15]采用了基于分叉树图的规划方法, 定义$\rho = \sum\limits_{i=1}^{n} 1/N^i_{\max}$为系统的传输密度, 则其中的引理3.3可重写为:

引理 3 若所有进程周期$N^i_{\max}$均为$x2^y$的形式$(x > 0, x, y \in \mathbb{N})$, 则一定能在总数为$\varrho = \lceil \rho \rceil$的信道内被规划.

基于上述引理, 文献[16]在提出了分组规范周期的方法, 将周期变为$N^i = x2^y \leqslant N^i_{\max}$的形式, 并论证了算法占用的信道数量上限为$\varrho + 2\sqrt{\varrho}$, 流程描述如下:

算法 2 周期长度规范化.

输入: 最大周期$N^i_{\max}$.

输出: 规范周期N^i, 分组编号set^i.

1) 计算分组总数$\mathrm{num} = \lceil \sqrt{\varrho} \rceil$.

2) 遍历$k = 1, 2, \cdots, \mathrm{num}$, 计算$y^k = \max\{y \in \mathbb{N} : (2k-1)2^y \leqslant N^i_{\max}\}$, 令$N^i_k = (2k-1)2^{y_k}$.

3) 将周期规范为$N^i = \max\limits_k (N^i_k)$, 即最接近$N^i_{\max}$的数值, 并将对应的$k$记为$\mathrm{set}^i$.

执行算法2后$N^i \leqslant N^i_{\max} < 2N^i$, 根据编号$\mathrm{set}^i$的数值, 进程将被分为num组分别进行序列规划.

例 1 对于最大周期为$\{1, 2, 3, 4, 5, 6\}$的进程$s_1 \sim s_6$, 分组数量$\mathrm{num} = 2$, 分组编号为$\mathrm{set}^{1,2,4,5} = 1$, $\mathrm{set}^{3,6} = 2$, 规范周期为$\{1, 2, 3, 4, 4, 6\}$, 信道总数$3 < 5.58(\varrho + 2\sqrt{\varrho})$.

为确保算法对在线规划的可拓展性, 本文生成序列的算法以文献[17]中缓存区规划的方法为基础. 为解释后续流程, 定义$\rho^s(k) = \sum\limits_{\mathrm{set}^i = k} 1/N^i$为分组传输密度, 则编号为$k$的进程组占用信道总数为$\lceil \rho^s(k) \rceil$. 因$\rho^s(k) \neq \lceil \rho^s(k) \rceil$意味着在一些时刻的不必完全占用信道, 而文献[17]在排序中却不允许空闲, 这会导致进程s_i的实际周期小于N^i, 使通讯频次超过所需, 加速传感器能量消耗. 因此, 引进了参与排序但不传输数据的空进程, 其流程描述如下:

算法 3 空进程引进.

输入: 进程集$\{s_i\}$, 规范周期集$\{N^i\}$, 分组编号集$\{\mathrm{set}^i\}$.

输出: 进程集$\{s_i\} \cup \{s_{\mathrm{e}}\}$, 周期集$\{N^i\} \cup \{N^{\mathrm{e}}\}$, 分组编号集$\{\mathrm{set}^i\} \cup \{\mathrm{set}^{\mathrm{e}}\}$.

1) 遍历$k = 1, 2, \cdots, \mathrm{num}$, 计算$\rho^s(k)$.

2) 若$\rho^s(k) \neq \lceil \rho^s(k) \rceil$, 检索组内进程周期的最大值$\bar{N}_k = \max\limits_{\mathrm{set}^i = k} N^i$.

3) 添加$\bar{N}_k \times (\lceil \rho^s(k) \rceil - \rho^s(k))$个周期为$\bar{N}_k$的空进程$s_{\mathrm{e}_1}, s_{\mathrm{e}_2}, \cdots$, 使等式成立, 改写周期和编号集.

因为同组进程的规范周期形式$(x2^y)$具备相同的$(x =)2k-1$的值, $\bar{N}_k$必然是同组周期的最小公倍数, 流程3)中的乘式的结果为整数.

例 2 例1中分组1的$\rho^s(1) = 2$, 无需空进程. 分组2的$\rho^s(2) = 0.5$, 同组最大周期$\bar{N}_2 = 6$, 添加了3个周期为6的空进程.

完成空进程的引进后, 对各进程组分别进行规划, 各组的规划流程如下:

算法 4 基于缓存的分组规划.

输入: 前述进程、周期和分组编号集.

输出: 规划序列λ.

1) 初始化, 遍历$k = 1, 2, \cdots, \mathrm{num}$, 建立缓存区$B_k$, 令序列$\lambda$中所有$\lambda^i_t = 0$.

2) 0时刻, $B_k = \{B_k(1), B_k(2), \cdots, B_k(\bar{N}_k)\}$, 将$\mathrm{set}^i = k$的进程$s_i$加入缓存块$B_k(N^i)$.

3) t时刻, 若进程$s_i \in B_k(1)$, 令传输标志$\lambda^i_t = 1$, 待流程6)之后将s_i重置到缓存块$B_k(N^i)$中.

4) 此时若$\sum\limits_{\mathrm{set}^i = k} \lambda^i_t < \lceil \rho^s(k) \rceil$, 即$B_k(1)$中原有进程数少于该组信道数量, 转流程5); 否则转流程6).

5) 遍历缓存块$B_k(j)$, $j \geqslant 2$. 对于$s_x \in B_k(j)$, 计算$d_i = N^i - j$并排序. 令d_i数值最大的$\lceil \rho(k) \rceil - \sum\limits_{\mathrm{set}^i = k} \lambda^i_t$个进程$s_x$的传输标志$\lambda^x_t = 1$, 并将$s_x$重置到缓存块$B_k(N^x)$中(数值相同时, N^x小的s_x优先). 转流程6).

6) 将$B_k(j)(j \geqslant 2)$中元素转入$B_k(j-1)$.

7) $t = t + 1$, 转流程3).

例 3 前例中$\mathrm{set}^i = 1$的进程组前4步规划如表1

所示, 优先序是按算法规则排列的进程择取排序, 因$\lceil\rho(k)\rceil=2$, 每时刻选择2个进程令相应的$\lambda_t^i=1$, 如规划栏所列.

表 1 离线规划实例

Table 1 The off-line scheduling example

t	B_1	优先序	规划
0	$\{s_1\},\{s_2\},\varnothing,\{s_4,s_5\}$	s_1,s_2,s_4,s_5	λ_0^1,λ_0^2
1	$\{s_1\},\{s_2\},\{s_4,s_5\},\varnothing$	s_1,s_4,s_5,s_2	λ_1^1,λ_1^4
2	$\{s_1,s_2\},\{s_5\},\varnothing,\{s_4\}$	s_1,s_2,s_5,s_4	λ_2^1,λ_2^2
3	$\{s_1,s_5\},\{s_2\},\{s_4\},\varnothing$	s_1,s_5,s_2,s_4	λ_3^1,λ_3^5

3.3 在线动态调整(On-line adjusting algorithm)

本节描述基于离线规划序列的在线动态调整方法, 由两部分组成: 空闲信道重发, 针对数据在传输中的随机错漏, 利用空闲进程所属信道发送新的测量; 序列重规划, 针对进程指标的变化, 改变传输周期和并调整传输序列.

首先介绍空闲信道重发机制. 当信号在无线传输过程中受扰出错, 滤波器无法获得测量. 此时进程的实际s_i传输间隔为$2N_i>N_{\max}^i$或更大, 导致误差协方差超标. 文献[14]设计了微调传输序列并利用空闲信道重发的方法, 其实现需要传感器的交互, 不适应网络实际; 其中对已超限进程置之不理的设计, 导致下次测量信息来临前估计的不断恶化. 针对这些不足, 本文调整了重发机制的规则. 假设传感器具备对离线传输序列中空闲信道的所有知识, 并假设传感器在时刻传输数据出错时能收到反馈, 最早在$t+1$时刻规划重发. 本文称按前述离线序列的传输为正点传输, 利用空闲新增的传输为临时传输, 网络各部分服从下述规则:

1) 传感器: 收到传输出错或阻塞反馈时, 检索空闲进程列表, 设最近的空闲时刻为$t_{\rm order}+\tau_{\rm e}$, 其中$t_{\rm order}$是传感器本次正点传输出错时刻, $\tau_{\rm e}\in\mathbb{N}^+$. 若$\tau_{\rm e}<N^i$, 在$t_{\rm order}+\tau_{\rm e}$时刻新增一次数据$y^i_{t_{\rm order}+\tau_{\rm e}}$的临时传输.

2) 中继节点: 无线网络的传输通常经过中继节点, 我们为中继节点设定了下述功能: a) 数据校验, 出现传输错误时将反馈传感器. b) 冲突避免, 设置传输前的握手流程, 选择传输优先级高的数据, 若某时刻所有传感器新增的临时传输数量超过空闲信道数量, 中继仅传输允许数量的临时传输($N_{\max}^i$最大优先), 并反馈被阻塞数据的源节点.

3) 控制中枢: 对于接收的临时传输数据$y^i_{t_{\rm order}+\tau_{\rm e}}$, 若$\tau_{\rm e}\leqslant N_{\max}^i-N^i$, 其进程误差协方差始终达标; 否则, 协方差将在下一临时或正点传输时刻前一定时间内超标. 若进程s_i的协方差出现超标, 设时刻$t_{\rm new}$是此后首次成功传输的时刻, 时刻$t_{\rm next}$是$t_{\rm new}$之后首次正点传输的时刻. 控制中枢判断$h_i^{t_{\rm next}-t_{\rm new}-1}(P_{t_{\rm new}})\leqslant P_{\rm sup}^i$不成立时, 通知传感器$i$在时间段$(t_{\rm new},t_{\rm next})$间新增一次临时传输.

注 2　临时传输分两类: 传感器自主发起(规则1))和控制中枢命令(规则3)). 中继节点的存在可协调自发传输引起的冲突.

当进程s_i的协方差指标改变时, 最大周期$N_{\max}^i$可能随之不同, 需要对重新规划传输序列, 类似的需求也发生于进程终止或新增时. 为避免对系统整体的初始化, 设计了两种操作, 删除与新建, 调整进程的正点传输序列. 删除对应于终止进程, 新建对应于新增进程, 进程变动是通过删除与新建结合. t时刻收到变动指令后, 操作流程如下:

算法 5　序列调整操作.

输入: 操作进程s_i, 进程集, 分组编号集, 周期集, 缓存区$B_1,B_2,\cdots,B_{\rm num}$.

输出: 进程集, 分组编号集集, 周期集, 缓存区$B_1,B_2,\cdots,B_{\rm num}$.

删除

1) 检索$s_i\in B_k(t),k=\mathrm{set}^i$, 将$s_i$从该缓存中删除.

2) 在$B_k(\bar N_k)$中添加$\bar N_k/N^i$个周期为$\bar N_k$的空进程, 并修改相应各集合.

新建

3) 遍历$k=1,2,\cdots,\mathrm{num}$, 据算法2流程2)计算$N_k^i$, 若$N_k^i>\bar N_k$, 令$N_k^i=\bar N_k$.

4) $N^i=\max\{N_k^i:\rho^s(k)+1/N_k^i\leqslant\lceil\rho^s(k)\rceil\}$, 记录相应$\mathrm{set}^i$.

5) 在缓存区$B_k,k=\mathrm{set}^i$中删除$\bar N_k/N^i$个空进程, 将s_i加入缓存块$B_k(N^i)$.

6) 若流程4)中不存在满足条件的N^i, 令$N^i=\max(N_k^i)$, 记录相应set^i并在该组中新增一条信道, 按算法3补全空进程.

为避免对空进程和缓存区进行初始化, 上述流程4)限制了N_k^i的大小. 上述调整主要针对缓存区进行操作, 新序列仍由算法4生成. 上述新建操作在调整后初期或将造成一些时刻$B_k(1)$中元素数量超过该组信道总数$\lceil\rho^s(k)\rceil$, 在规划中优先选择N^i最小的$\lceil\rho^s(k)\rceil$个进程, 并将$B_k(1)$中所有进程按算法4流程3)重置, 序列将在$\bar N_k$个时刻内恢复正常.

例 4　设进程$s_1\sim s_6$具备规范周期$\{4,4,8,8,8,8\}$, 其中$s_3\sim s_6$是空进程, t时刻原序列为$\{s_1,s_2,s_3,s_4,s_1,s_2,s_5,s_6,\cdots\}$, 收到指令添加$N^7=2$的新进程$s_7$后新序列为$\{s_1,s_7,s_1,s_7,s_2,s_7,s_1,s_7,s_2,$

$s_7,\cdots\}$, 事实上自$t+2$时刻起序列已调整完毕, 其规划如表2所示.

表 2 序列调整操作实例

Table 2 The on-line scheduling example using Algorithm 5

t	B_1	优先序	规划
原	$\{s_1\},\{s_2\},\{s_3\},\{s_4\},\cdots$	$s_1,\cdots$	λ_t^1
新	$\{s_1\},\{s_2,s_7\},\varnothing,\varnothing,\cdots$	s_1,s_7,s_2	λ_t^1
+1	$\{s_2,s_7\},\varnothing,\varnothing,\{s_1\},\cdots$	s_7,s_2,s_1	λ_{t+1}^7
+2	$\varnothing,\{s_7\},\{s_1\},\{s_2\},\cdots$	s_1,s_7,s_2	λ_{t+2}^1

4 实验结果与分析(Simulations)

本节中将对前述算法进行仿真验证, 分析本文算法效果, 并与文献[14]算法进行比较.

首先比较周期计算方法, 对于参数为$A^{(1)}=1.05$, $C^{(1)}=1$, $Q^{(1)}=1$, $R^{(1)}=4$的进程s_1, 随机生成1000组协方差指标 $P_{\rm sup}^{(1)}$=X+$\check{P}_1^{(1)}$, 其中 X 为区间[0, 100]中的随机数. 采用本文算法计算的周期平均值14.66, 文献[14]算法计算的周期平均值为13.40. 可见, 相比于文献[14] 中基于协方差上界的计算, 本文中以更多计算为代价获得更长的传输周期, 进一步降低了网络通讯负荷. 需要指出的是, 对于参数为$A^{(2)}=[1,0.1;0,1]$, $C^{(2)}=[1,0]$, $Q^{(2)}=[4,0;0,1]$, $R^{(2)}=9$的 进程s_2, 文献[14] 所用的更新后协方差上界为[9, 90; 90, 2201](根据文献[18]备注4.8的推导计算), 而事实上周期较大时滤波器更新后的协方差也不足以与之比拟, 例见$\check{P}_{100}^{(2)}=[8.99,1.05;1.05,35.92]$, 前一个上界在周期计算中显然是没有意义的. 因此文献[14]的周期计算对滤波器参数有更多限制, 而本文方法则更具普适性.

为验证本文周期计算过程中夹挤算法的作用, 针对前述进程s_1和s_2计算满足1000组不同指标的周期. 其中: s_1 的指标 $P_{\rm sup}^{(1)}$ 随机生成规则同上, s_2 的指标 $P_{\rm sup}^{(2)}$=diag$\{X,Y\}+\check{P}_1^{(2)}$, X,Y分别为区间$[0,1000]$, $[0,50]$ 中的随机数. 为简化比较, 笔者统计了运算过程中计算稳态协方差$\check{P}_k^i$的次数, 显然通过逐步遍历的方法计算出最大周期N需要计算$\check{P}_k^i$ 的次数与周期长度相等. 上述s_1和s_2 以两种不同的方法的平均计算次数对比如表3所示.

表 3 $\check{P}_k^i$的平均计算次数

Table 3 Average number of calculations for $\check{P}_k^i$

进程	逐步遍历	夹挤算法
s_1	14.66	3.01
s_2	14.14	2.62

尽管$\check{P}_k^i$的计算次数的对比不足以反映实际计算量的差别, 但其中悬殊的比例却足以说明, 利用夹挤算法将显著减小检索步数, 提高计算效率.

为比较本文和文献[14]中规划算法的差异, 随机生成进程和相关指标进行实验. 表4为不同数量进程分别以本文和文献[14]规划, 所需信道数量的对比, 进程传输密度ρ也附于表中, 各组数据均为100次实验的平均值. 为了明确对比并适应实际, 实验中进程的最大传输周期计算均以本文方法实现, 并筛选了周期不超过20的进程进行实验. 实验结果表明, 本文算法比文献[14] 中的扩展算法占据更少信道, 进程数量越多差值越明显.

表 4 平均传输密度和信道数量

Table 4 Average transmission density and channel numbers

进程数	传输密度	本文信道数	文献[14]信道数
20	2.68	3.81	3.82
30	4.23	5.67	5.80
40	5.55	7.12	7.47
50	6.91	8.66	9.37

为评价算法效果, 对前述生成的进程进行时长为10000步的实验, 统计计算了每个进程超过协方差阈值的平均步数, 实验中数据的传输是存在不同概率的错漏的. 图2是前述进程数量为30的100个实验组在不同传输错漏概率下的平均实验结果. 笔者假设, 传输错漏是服从伯努利分布的. 实验结果表明, 概率增大时更容易引起传输失败使协方差超限, 导致进程平均超限时间增大. 运行时额外增加传输信道有助于减少进程的超限时间, 提高系统对高传输错漏率的适应性. 文献[14]扩展算法的效果也绘制在图中, 与其相比本文规划算法的平均超限时间更短.

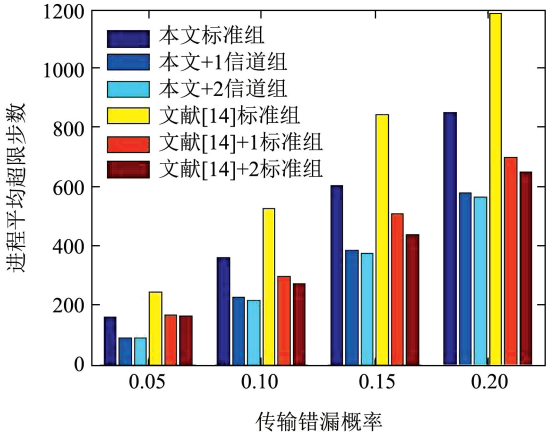

图 2 本文与文献[14]算法中各进程的平均超限步长(总时长10000步)

Fig. 2 Average number of over-threshold steps in processes of the paper and Ref.14 (total length: 10000 steps)

为了直观显示本文和文献[14]中规划算法的差异,

本文选择了实验中的一个典型进程进行对比, 如图3所示. 图中的进程参数与s_1相同, 初始时协方差指标为$P_{\rm sup}=12$. 区分所需, 记基于本文和文献[14]中算法所规划序列的进程符号为$s^{\rm the}$和$s^{\rm ref}$, 其协方差为$P_t^{\rm the}$和$P_t^{\rm ref}$. 图中时间开始时的周期为$N_{\max}^{\rm the}=6$, $N^{\rm the}=6$, $N_{\max}^{\rm ref}=5$, $N^{\rm ref}=4$(与前述实验中不同, 这里进程周期分别以本文和文献[14]方法计算). $t=300$时刻, 协方差指标变为$\hat{P}_{\rm sup}=18$, 则周期变为$\hat{N}_{\max}^{\rm the}=8$, $\hat{N}^{\rm the}=8$, $\hat{N}_{\max}^{\rm ref}=8$, $\hat{N}^{\rm ref}=8$. 数据成功传输时, 滤波器根据测量信息进行后验估计, 误差协方差减小; 否则无测量信息, 滤波器只进行先验估计, 误差协方差增大: 因此图中协方差呈波浪形. 传输错漏的存在, 破坏了规划序列的严格周期性, 一些新的数据传输被规划, 其时间点如图中“o”符号所标注.

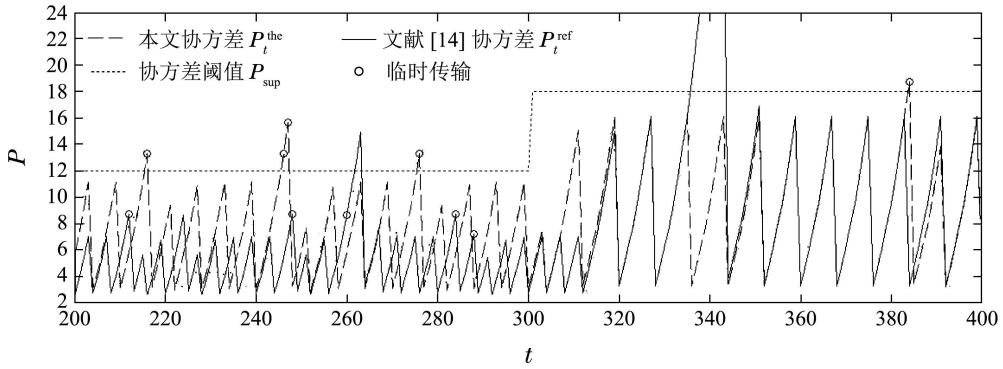

图 3 本文与文献[14]算法的协方差对比

Fig. 3 The error covariance of algorithms in the paper and Ref. [14]

根据图3, 不考虑传输错漏的情况下, 本文进程协方差的峰值更接近阈值, 在保证估计精度时更能有效减少传输次数. 出现传输错漏时, 与文献[14]中超限后不再新增传输的设计相比, 本文对空闲信道的利用更为灵活, 这正是上图2中本文算法进程平均超限时间更短的原因. 在图3中, 在$t=245, 246$时刻$s^{\rm the}$连续发生传输错误, 本文算法空闲信道新增了两次传输; 而在$t=259, 260$时刻$s^{\rm ref}$也发生了连续传输错误, 根据文献[14]的设计$t=261$时刻起不再规划新的传输, 下次传输前一定时间内协方差将持续超限. 类似的对比发生于$s^{\rm the}$的$t=215, 275$, 时刻和$s^{\rm ref}$的$t=335$时刻.

此外, 本文的传输周期调整算法响应时间更短, 周期长度调节更灵活. 在$t=300$时刻, 进程指标变化, $s^{\rm the}$在$t=304$时刻完成调整, $s^{\rm ref}$的算法存在滞后, 其周期在$t=312$时刻调整. 本文和文献[14]中规范化周期均为$x2^y$, $x=2k-1$形式, 不同之处在于本文中存在多个基值k, 文献[14]初始化时k 已固化, 显然指标变化时更多基值k的选择, 能使最大周期与规范周期更为接近. 例如, $s^{\rm the}$的规范周期原以$k=2$为基础, 后以$k=1$为基础, 与最大周期始终保持一致, 其数据传输次数比$s^{\rm ref}$更少.

5 结语(Conclusions)

本文针对并行卡尔曼滤波器的序列规划, 提出论证了快速计算进程周期的夹挤算法, 设计了基于多信道的序列规划与调整算法, 以满足各滤波器对应进程的协方差指标. 仿真实验表明, 本文算法显著减少了周期计算的消耗; 进程的传输周期更接近其最大值, 降低了通讯消耗; 对空闲信道利用更充分, 有助于缩短错误导致的协方差超限总时间, 并降低误差峰值; 针对协方差指标变化的调整时间更短.

参考文献(References):

[1] MUSICK S, MALHOTRA R. Chasing the elusive sensor manager [C] //*Proceedings of IEEE National Aerospace and Electronics Conference*. Dayton, America: IEEE, 1994, 1: 606 – 613.

[2] GUPTA V, CHUNG T, HASSIBI B, et al. Sensor scheduling algorithms requiring limited computation [C] //*Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*. Montreal, Canada: IEEE, 2004, 3: 825 – 828.

[3] SANDBERG H, RABI M, SKOGLUND M, et al. Estimation over heterogeneous sensor networks [C] //*IEEE Conference on Decision and Control*. Cancun, Mexico: IEEE, 2008: 4898 – 4903.

[4] MO Y, GARONE E, SINOPOLI B. On infinite-horizon sensor scheduling [J]. *Systems & Control Letters*, 2014, 67(7): 65 – 70.

[5] ORIHUELA L, BARREIRO A, GÓMEZ-ESTERN F, et al. Periodicity of Kalman-based scheduled filters [J]. *Automatica*, 2014, 50(10): 2672 – 2676.

[6] SINOPOLI B, SCHENATO L, FRANCESCHETTI M, et al. Kalman filtering with intermittent observations [J]. *IEEE Transactions on Automatic Control*, 2004, 49(9): 1453 – 1464.

[7] ROHR E, MARELLI D, FU M. On the error covariance distribution for kalman filters with packet dropouts [J/OL]. *Discrete Time Systems*, 2011: 71 – 92. http://www.intechopen.com/books/discrete-time-systems/on-the-error-covariance-distribution-for-kalman-filters-with-packet-dropouts.

[8] XIAO Li, SUN Zhigang, HU Xiaoya, et al. Remote state estimation based on sensor networks [J]. *Control Theory & Applications*, 2009, 26(7): 763 – 766.
(肖力, 孙志刚, 胡晓娅, 等. 基于传感器网络的远程状态估计 [J]. 控制理论与应用, 2009, 26(7): 763 – 766.)

[9] LIU Yonggui, XU Bugong, SHI Buhai. Kalman filtering for stochastic systems with consecutive packet losses and measurement time delays [J]. *Control Theory & Applications*, 2013, 30(7): 898 – 908.
(刘永桂, 胥布工, 史步海. 随机系统中能容忍连续丢包和测量时延的卡尔曼滤波 [J]. 控制理论与应用, 2013, 30(7): 898 – 908.)

[10] REN Z, CHENG P, CHEN J, et al. Optimal periodic sensor schedule for steady-state estimation under average transmission energy constraint [J]. *IEEE Transactions on Automatic Control*, 2013, 58(12): 3265 – 3271.

[11] LI Y, QUEVEDO D E, LAU V K N, et al. Optimal periodic transmission power schedules for remote estimation of ARMA processes [J]. *IEEE Transactions on Signal Processing*, 2013, 61(24): 6164 – 6174.

[12] SHI L, JOHANSSON K H, QIU L. Time and event-based sensor scheduling for networks with limited communication resources [C] //*Proceedings of World Congress of the International Federation of Automatic Control*. Milano, Italy: IFAC, 2011, 44(1): 13263 – 13268.

[13] LIN Z, WANG C. Scheduling parallel Kalman filters for multiple processes [J]. *Automatica*, 2013, 49(1): 9 – 16.

[14] LU S, LIN Z, ZHENG R, et al. Scheduling parallel Kalman filters with quantized deadlines [J]. *Systems & Control Letters*, 2015, 86: 9 – 15.

[15] BAR-NOY A, LADNER R E. Windows scheduling problems for broadcast systems [J]. *Siam Journal on Computing*, 2003, 32(4): 433 – 442.

[16] BAR-NOY A, LADNER R E, TAMIR T. Windows scheduling as a restricted version of Bin packing [J]. *ACM Transactions on Algorithms*, 2007, 3(3): 224 – 233.

[17] BAR-NOY A, LADNER R E, CHRISTENSEN J, et al. A general buffer scheme for the windows scheduling problem [J]. *Journal of Experimental Algorithmics*, 2009, 13: 6 – 17.

[18] SHI L, EPSTEIN M, MURRAY R M. Kalman filtering over a packet-dropping network: a probabilistic perspective [J]. *IEEE Transactions on Automatic Control*, 2010, 55(3): 594 – 604.

作者简介:

陆诗远 (1988–), 男, 博士研究生, 研究方向为无线传感网络的调度与优化, E-mail: lushiyuan@zju.edu.cn;

颜钢锋 (1959–), 男, 教授, 博士生导师, 研究方向为无线传感网络、机器人和混杂系统控制, E-mail: ygf@zju.edu.cn;

林 澈 (1989–), 男, 博士研究生, 研究方向为分布式卡尔曼滤波和网络协同定位, E-mail: linche@zju.edu.cn;

郑荣濠 (1984–), 男, 讲师, 博士, 研究方向为多智能体系统的协同控制与优化, E-mail: rzheng@zju.edu.cn.