RFID 系统中面向服务的动态资源配置优化机制

刘建华，童维勤
(上海大学 计算机工程与科学学院，上海 200072)

摘 要：针对物联网中 RFID(radio frequency identification)系统的 tag 和 reader 的速率不匹配及其资源受限问题，从理论上对 tag 和 reader 之间的协作通信上下文进行分析，提出一种面向服务的 RFID 系统资源配置优化机制，该机制利用 tag 与 reader 的组件 QoS(quality of service)信息建立闭环控制框架，使用 3 种控制策略根据其资源状态和类型调节 QoS 参数。通过实验对机制进行了性能分析和验证，结果表明，该优化机制有效地提高 RFID 系统的吞吐量，高效地使用系统资源，减少了能量消耗和 reader-to-reader 干扰。

关键词：物联网；RFID 无线传感网络；资源配置；模糊逻辑控制
中图分类号：TP393 **文献标识码**：A **文章编号**：1000-436X(2012)08-0093-13

Service-oriented dynamic resource configuration optimization scheme in RFID system

LIU Jian-hua, TONG Wei-qin
(School of Computer Science and Engineering, Shanghai University, Shanghai 200072, China)

Abstract: Aiming at the defect that the rate between tag and reader was not match and their resource-constrained problem in the radio frequency identification (RFID) systems, the collaboration communication context was analyzed theoretically. Service-oriented dynamic resource configuration optimization scheme of RFID system was proposed which used the component quality of service(QoS) information of tag and reader to establish closed-loop control framework and developed three control strategies to adjust QoS parameter according to the state and type of its resources.The scheme was analyzed and validated for performance through experiments.The numerical experiments results show that the optimization mechanism make throughput of RFID systems improve, use resources efficiently and reduce the energy consumption and interference of reader-to-reader.

Key words: internet of thing; RFID sensor network; resource configuration; fuzzy logic control

1 引言

物联网是下一代网络的发展方向，RFID 作为物联网的一大支撑技术已经大量应用于环境中的对象识别[1~4]。然而，这些已经部署的系统适用于小规模的应用，当进行大规模的开发和应用 RFID 系统时，就会面临许多的资源优化问题[5],特别是智能电子标签技术是物联网高效应用的瓶颈，如何设计低成本、低功耗的电子标签是物联网应用面临的巨大挑战，其次还面临着电子标签覆盖，能量高效及干扰优化等关键问题，MC-BFO[6]算法集成了 Cell-to-Cell 通信方法对 RFID network 进行了优化。本文从资源优化角度出发，为下一代智能电子标签和读写器的设计提出了面向服务的 RFID sensor 系

收稿日期：2011-07-05；**修回日期**：2011-10-05
基金项目：上海市重点学科基金资助项目（J50103）；上海大学创新基金资助项目（SHUCX101061）
Foundation Items: Shanghai Leading Academic Discipline Project (J50103) ; Graduate Innovation Foundation of Shanghai University (SHUCX101061)

统动态资源配置方法，此方法提供了一个在 RFID 系统中的资源配置机制,它是利用资源分配和 RFID tag 和 reader 内部服务调整来实现 RFID 应用系统的吞吐量管理的一种重要手段，同时也是在传感节点软件动态更新后，对其上的资源优化配置所必需的服务。在物联网环境中，由于 RFID sensor 资源缺乏，如果不进行综合性的资源配置优化，RFID 系统很难高效的应用，单一的资源配置是一种局部的解决方案，适用于特定的场合，难以满足日益增多的 RFID sensor 系统。此外，当前无线传感器技术还难以消除移动给 RFID 系统带来的链接不稳定状态，主要表现在当贴在车辆上的 RFID 电子标签随车移动以及手持读写器随人移动时，带宽就会发生变化，引起了 RFID 系统吞吐量起伏不定。例如在一个车辆稽查系统中，稽查员手持读写器读取过往车辆上的电子标签，这些电子标签数据经过预处理（过滤、压缩、加密），然后通过无线网络传输到车辆控制中心以供车辆稽查。另外，在一些水上漂浮的传感器，当获得其数据并存储在 RFID tag 中后，同样也由于移动影响其快速链接到 RFID reader，在这些应用场景中，有时还有干扰、速率不匹配等情况发生。资源配置服务的关键问题是，如何使 RFID 系统在带宽波动、内存、电池、MCU、CPU 计算能力有限的情况下提供满意的吞吐量。近年来，一些学者在应用资源配置服务方法对系统性能优化方面做了深入研究工作，这其中有基于组件的嵌入式软件 QoS 分析方法，为了解决多准则优化问题，一个基于组件选择的方法根据多个准则使用了权重方程给出每个组件的总效用值,也有一些方法使用预测系统的 QoS 属性,例如资源消耗[7]和可靠性[8]等。RFID 设备嵌入式软件组件是集成在一个容器中，基于容器的嵌入式软件开发方法[9]为后来的嵌入式软件组件 QoS 优化[10]起到了基础作用。Li 等[11]提出了在移动网格中基于设备上下文的资源优化算法。对于系统的资源配置 Li 和 Nahrstedt[12]提出了基于 PID 的模糊控制模型来动态地分配 CPU 和带宽资源。Huang 等[13]为实时的多媒体服务开发了一个资源预留的模式去确保系统的 QoS。这些方法中的静态预留和动态分配很好地解决了多媒体系统的资源问题。

2 RFID 系统需要资源配置优化场景

图 1 中是 tag 和 reader 之间的速率不匹配场景，使用下文中的调节算法可以调节 tag 的发送速率；图 2 是读写器 2 干扰读写器 1 的场景,使用下文中的功率带宽联合算法可以调节读写器的功率，使得读写器 2 解除对读写器 1 的干扰。在对 RFID 系统中 reader-to-reader 干扰研究主要考虑 2 个重要的参数(SIR, signal-to-interference ratio)和功率，Mercado 等[14]为无线多媒体网络使用功率控制和智能天线提出了一个自适应的 QoS 概念，显示了对于多媒体用户在 SIR 平均等级下的数量增长。

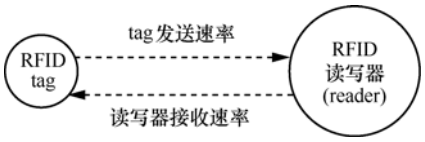

图 1 RFID tag 和 reader 速率不匹配

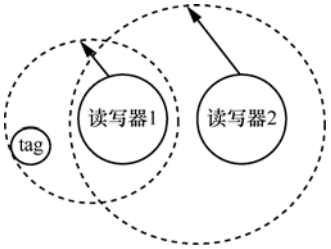

图 2 reader-to-reader 干扰

3 面向服务的资源消耗模型

3.1 建立 RFID 系统组件服务模型

RFID tag 服务 agent 是一个带有传感器的电子标签，它可以被 RFID 读写器读取，在工作状态下 RFID tag 服务 agent 会消耗其上的资源，同时执行特定功能将处理的 sensor 数据传输给 RFID reader 服务 agent，为了便于考查 RFID 系统的性能，主要关注其吞吐量（读取率）属性，给出一个 RFID 传感器服务 agent 的形式化定义如下。

定义 1 RFID 传感器服务 agent 可以表示为一个 5 元组：$F=\{D_{rs},C_{rs},I_{rs},O_{rs},Q_{rs}\}$。这里 D_{rs} 是关于 RFID 传感器的描述，其中包括附加其上传感器功能和状态，C_{rs} 是组件集合，I_{rs} 是该输入传感器相关的变量、数据格式和通信协议，O_{rs} 是该 RFID 传感器输出的相关变量、数据格式和通信协议等；Q_{rs} 是决定其吞吐量的 QoS 属性集合，例如，数据传输时间、MCU 使用率、内存消耗等。图 3 是一个 RFID 传感器功能服务图。包含有可视传感器，图像传感器(V_S)，非可视传感器，如温度、湿度(D_S)等。

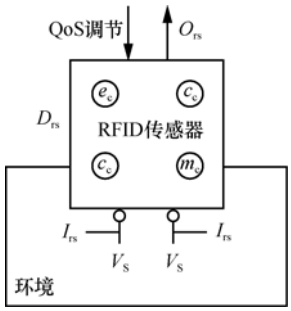

图 3　RFID 传感器节点功能服务

RFID tag 和 reader 及附着其上的 sensor 共同称作 RFID 传感器。RFID sensor 的配置函数如下：$conf_c^i(z_c)$ 是一个组件 z_c 的 QoS 属性值 a^i 配置函数，即由参数 z_c 对应的一个 QoS 属性值 a^i 的值 v_i^c。e_c 表示能量调节组件，c_g 表示通道调节组件，c_c 表示 CPU 调节组件，m_c 表示存储调节组件。

3.2　建立面向服务的资源优化分析模型

3.2.1　面向服务的 RFID 系统资源消耗模型

出于对资源的动态配置，提出了一个基于服务 agent 的 RFID 应用系统，如图 4 所示。RFID reader 服务 agent 一方面向资源服务 agent 提出申请资源并获得提供标签处理、数据存储、能量的资源；另一方面，发布自己的标签处理、数据存储、能量的资源服务，使得尽快识别 RFID tag，读取其数据。

图 4　面向服务的 RFID 系统资源消耗模型

RFID tag 服务 agent，一方面，向资源服务 agent 提出申请资源并获得提供标签数据写入、读出、传输、处理的能量及存储资源；另一方面，RFID tag 服务 agent 主动发布自己的能量资源，使得尽快获得 RFID reader 服务 agent 提供的标签处理、数据存储、能量的资源服务。资源服务 agent 一方面接受 RFID reader 服务 agent，RFID tag 服务 agent 的资源申请；另一方面，通过资源评估后分别向 RFID Reader 服务 agent 和 RFID tag 服务 agent 提供资源。资源上下文表示如下：

```
<Service>
<RFID System-Context>
  <Network-Context>
    <Wireless-Bandwidth>8MHz</Wireless-
Bandwidth
    <Connection-Rate>1Mbit/s</Connection-
Rate>
  </Network-Context>
  <Tag-Context>
    <MCU>100MHZ</MCU>
    <Available-Memory>256k</Available-
Memory>
    <Residual-Battery>40%</Residual-Battery>
  </Tag-Context>
  <Reader-Context>
    <CPU>667MHz</CPU>

<Available-Memory>256M</Available-Memory>
    <Residual-Battery>50%</Residual-Battery>
  </Reader-Context>
</RFID System-Context>
</Service>
```

设有 RFID tag 服务 agent $T_g=\{T_{g1},\cdots,T_{gn}\}$ 和 RFID reader 服务 agent $R=\{R_{d1},\cdots,R_{dm}\}$，$B_i^{n,j}$ 表示 T_{gi} 从 R_{dj} 获得的带宽 n，$M_i^{o,j}$ 表示 Tgi 从 R_{dj} 获得 *TagData* 的存储 o，$C_i^{p,j}$ 表示 T_{gi} 从 R_{dj} 获得处理 *TagData* 的 CPU p，$E_i^{q,j}$ 表示 T_{gi} 从 R_{dj} 获得处理和传输 *TagData* 的能量 q，$B_S^{n,j}$ 表示 R_{dj} 的总带宽容量，$M_S^{o,j}$ 表示 R_{dj} 的总存储容量，$C_S^{p,j}$ 表示 R_{dj} 最大 CPU 处理能力，$E_S^{q,j}$ 表示 R_{dj} 总的电池能量，$t_i^{r,j}$ 表示 R_{dj} 接收和处理 T_{gi} TagData 花费的时间，$B_i^{n,j,C}$ 表示 R_{dj} 接收 T_{gi} 花费的带宽，$M_i^{o,j,C}$ 表示 R_{dj} 存储 T_{gi} 花费的存储空间，$C_i^{p,j,C}$ 表示 R_{dj} 处理 T_{gi} 花费的 CPU，$E_i^{q,j,C}$ 表示 R_{dj} 接收和处理 T_{gi} *TagData* 花费的电池能量，$B_i^{n,j,T}$ 表示 T_{gi} 发送数据花费的带宽，$M_i^{o,j,T}$ 表示 T_{gi} 存储 Sensors 数据花费的存储空间，$C_i^{p,j,T}$ 表示 T_{gi} 处理数据花费的 MCU。$E_i^{q,j,T}$ 表示 T_{gi} 发送和存储数据花费的能量，$C_{i,S}^{(n,o,p,q),j}$ 表示 T_{gi} 接收、存储、处理 Sensors 数据总花费，T_l^j 表示 R_{dj} 接收和处理其读取半径内的 $T_{gi}(d_{ij}\leqslant r_i+r_j)$花费的时间。

在 RFID 服务网络中由多个 RFID tag sensor 组成了一个集合，它们可以用一个服务 RFID tag sensor 集合来表示 $y_{tg}=\{t_{g1},t_{g2},\cdots,t_{gn}\}$，假设用 $T_g(t_{gk})$

来表示 $t_{gk}(k\in[1,n])$服务区域内提供服务的 RFID tag sensor 集合，那么提供服务的 RFID tag sensor 消耗的电池能量总和为

$$E_{t_g}^{S}(t_{gk})=\sum_{t_{gi}\in T_g(t_{gk}),t_{gk}\in y_{tg}} E_{t_g}(t_{gi})$$

存储消耗总和为

$$M_{t_g}^{S}(t_{gk})=\sum_{t_{gi}\in T_g(t_{gk}),t_{gk}\in y_{tg}} M_{t_g}(t_{gi})$$

带宽消耗总和为

$$B_{t_g}^{S}(t_{gk})=\sum_{t_{gi}\in T_g(t_{gk}),t_{gk}\in y_{t\,g}} B_{t_g}(t_{gi})$$

RFID reader sensor 服务集合表示为 $y_{rd}=\{r_{d1}, r_{d2},\cdots,r_{dm}\}$,假设用 $R_d(r_{dk})$ 来表示 $r_{dk}(k\in[1,m])$ 服务区域内提供服务的 RFID reader sensor 集合，那么提供服务的 RFID reader sensor 消耗的电池能量总和为

$$E_{r_d}^{S}(r_{dk})=\sum_{r_{di}\in R_d(r_{dk}),r_{dk}\in y_{rd}} E_{r_d}(r_{di})$$

存储消耗总和为

$$M_{r_d}^{S}(r_{dk})=\sum_{r_{di}\in R_d(r_{dk}),r_{dk}\in y_{rd}} M_{r_d}(r_{di})$$

带宽消耗总和为

$$B_{r_d}^{S}(r_{dk})=\sum_{r_{di}\in R_d(r_{dk}),r_{dk}\in y_{rd}} B_{r_d}(r_{di})$$

一个 RFID tag sensor 的资源动态配置是通过 RFID tag 服务 agent 及其相关的 QoS 参数决定的。使得 $Q(T_{gi},t)=y(m_0^t,m_1^t,\cdots,m_p^t)$，这里 $m_p{}^t$ 是在 t 时刻时 QoS 参数值，在 RFID 系统中调节 RFID tag 服务 agent 的输出。在有多个 RFID tag 服务 agent 的监控环境中，单个 RFID tag 服务 agent 的数据读取率依赖于多个 RFID tag 服务 agent 组成的读取区域，使用 $Q(y_{tg},t)=x(y(T_{g1},t),\cdots,y(T_{gp},t))$，这里 $T_{gi}\in y_{tg}$，$i\in[0,n]$，区域内所有 RFID tag 被读取的延迟时间为所有在监控区域内的标签平均延迟时间

$$Q_T(y_{tg},t)_{\text{delay}}=\frac{1}{p}\sum_{T_{gi}\in y_{tg}(t)} Q(T_{gi},t)_{\text{delay}} \qquad (1)$$

一个 RFID reader sensor 的资源动态配置是通过 RFID reader 服务 agent 及其相关的 QoS 参数决定的。使 $Q(R_{di},t)=y(m_0{}^t,m_1{}^t,\cdots,m_p{}^t)$,这里 $m_p{}^t$ 是在 t 时刻时 QoS 参数值。在 RFID 系统中调节 RFID reader 服务 agent 的输出。

3.2.2 RFID 系统资源的效用优化

资源效用优化方法是给 RFID 系统分配资源使 RFID 网络有好的服务质量，以便在 RFID tag 和 RFID reader 的资源约束下最大化系统的效用 $U_{\text{RFID-System}}$，使用非线性最优化理论，在 RFID 系统中效用最优化形式化为

$$Maximize\,U_{\text{RFID-System}}=\sum_{i=1}^{S}U_i(B_i^{n,j},M_i^{o,j},C_i^{p,j},E_i^{q,j}) \qquad (2)$$

$$\begin{aligned}\text{Subject to }&\sum B_i^{n,j}\leqslant B_S{}^{n,j},\sum M_i{}^{o,j}\leqslant M_S{}^{o,j},\\&\sum_{n}^{n}C_i^{p,j}\leqslant C_S{}^{p,j},\sum_{p}^{o}E_i^{q,j}\leqslant E_S{}^{q,j}\\&\sum_{p}^{p}t_i^{r,j}\leqslant T_i{}^{j},\sum_{q}^{q}B_i^{n,j,T}+\sum_{n}^{n}M_i{}^{o,j,T}+\\&\sum_{p}^{r}C_i^{p,j,T}+\sum_{q}^{n}E_i^{q,j,T}\leqslant C_{i,S}^{o(n,o,p,q),j}\end{aligned} \qquad (3)$$

Lagrangian 方法能解决约束最优问题，使用这个方法解决 RFID system service agent 最优问题，下面应用 Lagrangian 方法去解这个方程。

$$\begin{aligned}L=&\sum_{i=1}^{S}U_i(B_i^{n,j},M_i^{o,j},C_i^{p,j},E_i^{q,j})-\lambda_i\left(\sum_{n=1}^{N}B_i^{n,j}-B_S^{n,j}\right)-\\&\beta_i\left(\sum_{o=1}^{O}M_i^{o,j}-M_S^{o,j}\right)-\varphi_i\left(\sum_{p=1}^{P}C_i^{p,j}-C_S^{p,j}\right)-\\&\mu_i\left(\sum_{q=1}^{Q}E_i^{q,j}-E_S^{q,j}\right)-\alpha_i\left(\sum_{r=1}^{R}t_i^{r,j}-T_i^{j}\right)-\gamma_i\left(\sum_{n=1}^{N}B_i^{n,j,T}+\right.\\&\left.\sum_{o=1}^{O}M_i^{o,j,T}+\sum_{p=1}^{P}C_i^{p,j,T}+\sum_{q=1}^{Q}E_i^{q,j,T}-C_{i,S}^{(n,o,p,q),j}\right)\end{aligned} \qquad (4)$$

其中，λ_i、β_i、φ_i、μ_i 是 T_{gi} 单位时间内获得 R_{dj} 带宽、存储器、CPU、电池能量。α_i、γ_i 是 T_{gi} 花费的总时间的时间单位和一个单位时间内的资源消耗。T_{gi} 的资源约束下的最大化效用 $U_{\text{RFID-Tg}}$，使用非线性最优化理论，在 RFID 系统中 $U_{\text{RFID-Tg}}$ 效用最优化形式化为

$$\begin{aligned}Maximize\,U_{\text{RFID-Tg}}=&\left(\sum_{r=1}^{R}t_i^{r,j}-T_i^{j}\right)-\\&\left(\sum_{n=1}^{N}B_i^{n,j,T}+\sum_{o=1}^{O}M_i^{o,j,T}+\sum_{p=1}^{P}C_i^{p,j,T}+\sum_{q=1}^{Q}E_i^{q,j,T}-C_{i,S}^{(n,o,p,q),j}\right)\end{aligned} \qquad (5)$$

$$\begin{aligned}\text{Subject to }&\sum_{r}t_i^{r,j}\leqslant T_i^{j},\sum_{n}B_i^{n,j,T}+\sum_{o}M_i^{o,j,T}+\\&\sum_{p}C_i^{p,j,T}+\sum_{q}E_i^{q,j,T}\leqslant C_{i,S}^{(n,o,p,q),j}\end{aligned} \qquad (6)$$

R_{dj} 的资源约束下的最大化效用 $U_{\text{RFID-Rd}}$，使用非线性最优化理论，在 RFID 系统中 $U_{\text{RFID-Rd}}$ 效用最

优化形式化为

$$MaxU_{\text{RFID-Rd}}=\sum_{i=1}B_i^{n,j,T}\log B_i^{n,j}+M_i^{o,j,T}\log M_i^{o,j}+C_i^{p,j,T}\log C_i^{p,j}+E_i^{q,j,T}\sqrt[4]{E_i^{q,j}} \tag{7}$$

$$\text{Subject to } \sum_n B_i^{n,j}\leqslant B_S^{n,j},\sum_o M_i^{o,j}\leqslant M_S^{o,j},\sum_p C_i^{p,j}\leqslant C_S^{p,j},\sum_q E_i^{q,j}\leqslant E_S^{q,j} \tag{8}$$

$$B_i'^{n,j}=B_S^{n,j}\frac{B_i^{n,j,T}}{B_i^{n,j,C}},M_i'^{o,j}=M_S^{o,j}\frac{M_i^{o,j,T}}{M_i^{o,j,C}} \tag{9}$$

$$C_i'^{p,j}=C_S^{p,j}\frac{C_i^{p,j,T}}{C_i^{p,j,C}},E_i'^{q,j}=E_S^{q,j}\frac{E_i^{q,j,T}}{E_i^{q,j,C}} \tag{10}$$

其中，$M_i^{k,T}$是电子标签完成第 k 个 sensor/tag 数据传送时的内存需求；$E_i^{k,T}$是电子标签完成第 k 个 sensor/tag 数据传送时的能量需求；$C_i^{k,T}$是电子标签完成第 k 个 sensor/tag 数据传送时的处理器需求；$B_i^{k,T}$是电子标签完成第 k 个 sensor/tag 数据传送时的带宽需求。因此，完成第 k 个 sensor/tag 数据传送时的完成时间为

$$t_i^k=\frac{B_i^{k,T}B_i^{n,j,C}}{B_S^{n,j}B_i^{n,j,T}}+\frac{M_i^{k,T}M_i^{o,j,C}}{M_S^{o,j}M_i^{o,j,T}}+\frac{C_i^{k,T}C_i^{p,j,C}}{C_S^{p,j}C_i^{p,j,T}}+\frac{E_i^{k,T}E_i^{q,j,C}}{E_S^{q,j}E_i^{q,j,T}} \tag{11}$$

$$MaxU_{\text{RFID-tg}}=\left(T_i^j-\sum_{k=1}^{K}\left(\frac{B_i^{k,T}B_i^{n,j,C}}{B_S^{n,j}B_i^{n,j,T}}+\frac{M_i^{k,T}M_i^{o,j,C}}{M_S^{o,j}M_i^{o,j,T}}+\frac{C_i^{k,T}C_i^{p,j,C}}{C_S^{p,j}C_i^{p,j,T}}+\frac{E_i^{k,T}E_i^{q,j,C}}{E_S^{q,j}E_i^{q,j,T}}\right)\right)-\left(\sum_{n=1}^{N}B_i^{n,j,T}+\sum_{o=1}^{O}M_i^{o,j,T}+\sum_{p=1}^{P}C_i^{p,j,T}+\sum_{q=1}^{Q}E_i^{q,j,T}-C_{i,S}^{(n,o,p,q),j}\right) \tag{12}$$

解 Lagrangian 方程[11]得

$$M_i^{o,j,T}=\left(\frac{M_i^{o,j,C}M_i^{k,T}}{M_S^{o,j}}\right)^{1/2}\frac{\sum_{k=1}^{k}\left(M_i^{o,j,C}M_i^{k,T}\right)^{1/2}}{T_i^j} \tag{13}$$

$$B_i^{n,j,T}=\left(\frac{B_i^{n,j,C}B_i^{k,T}}{B_S^{n,j}}\right)^{1/2}\frac{\sum_{k=1}^{N}\left(B_i^{n,j,C}B_i^{k,T}\right)^{1/2}}{T_i^j} \tag{14}$$

$$E_i^{n,j,T}=\left(\frac{E_i^{q,j,C}E_i^{k,T}}{E_S^{q,j}}\right)^{1/2}\frac{\sum_{k=1}^{N}\left(E_i^{q,j,C}E_i^{k,T}\right)^{1/2}}{T_i^j} \tag{15}$$

$$C_i^{p,j,T}=\left(\frac{C_i^{p,j,C}C_i^{k,T}}{C_S^{p,j}}\right)^{1/2}\frac{\sum_{k=1}^{N}\left(C_i^{p,j,C}C_i^{k,T}\right)^{1/2}}{T_i^j} \tag{16}$$

其中，$M_i^{o,j,T}$, $B_i^{n,j,T}$, $E_i^{q,j,T}$， $C_i^{p,j,T}$ 值为 tag 调节的下界。

从 RFID Reader 效用求得

$$B_i^{n,j}=\frac{B_i^{n,j,T}B_S^{n,j}}{\sum_n B^{n,j,T}} \tag{17}$$

$$M_i^{o,j}=\frac{M_i^{o,j,T}M_S^{o,j}}{\sum_o M_i^{o,j,T}} \tag{18}$$

$$C_i^{p,j}=\frac{C_i^{p,j,T}C_S^{p,j}}{\sum_p C_i^{p,j,T}} \tag{19}$$

$$E_i^{q,j}=\left(\frac{E_i^{q,j,T}E_S^{q,j}}{\sum_q E_i^{q,j,T}}\right)^{\frac{4}{3}} \tag{20}$$

其中，$B_i^{n,j}$，$M_i^{o,j}$ ，$C_i^{p,j}$，$E_i^{q,j}$为 reader 调节最优资源分配值的下界，为了最优资源配置，QoS 控制参数值要与这些效用值比较以获得 RFID 系统的最佳 QoS 平衡，即有 2 个作用：作为调节的期望；改变调节参数时作为其下界，判断是否越界。这样可以缩短调节步，很快达到资源配置的目标。

4　资源配置服务策略

当 RFID tag sensor 移动时，由于其数据传输的波动造成资源消耗的变化，如存储时间增长，延迟时间增长，导致电池的消耗也增长，这就决定了 RFID sensor 网络系统的关键 QoS 变化是不可避免的。此外，RFID 系统中 link 不如移动终端有极大的灵活性和自适应性，tag 和 reader 都不具备，因此，从 RFID 系统 link 的灵活性出发，设计和增强 RFID 系统的 link 自适应性，在多标签识别系统中，在 reader 识别范围内的标签会同时传输数据，每个 reader 的带宽被分成时间帧，每个帧由固定数量的时间隙组成，每个时间隙被使用去传输固定长度的 tag 标签或 sensor 数据分组，而 tag 数据在传输前分成固定大小的分组，在任何一个帧 n 中，有些最小的时间隙，这些最小的时间隙比正常的时间隙持续时间短，这预留给每个 tag 与 reader 之间交换控制消息。在 RFID 系统中的 tag 和 reader 之间的 QoS 控制要考虑以下 3 个方面：1)tag 期望的传输速率；2)reader 能给 tag 配置的资源；3)tag 和 reader 之间及其各自的 QoS

控制策略。资源配置服务的目标是通过 QoS 变化的有效管理，从而实现资源的有效利用和提高系统的吞吐量。为了实现这个目的，资源配置服务的 QoS 值低于阈值时，能够利用网络中的所有可用资源提高系统的吞吐量。相反，如果 QoS 值超出阈值时，要调低系统中 RFID sensor 消耗的 QoS 值，以增强网络的生命期。资源配置过程即为 QoS 调节过程。

在资源受限的 RFID 网络环境中，动态资源配置的目标是把 QoS 值控制到一个合理水平，这个控制过程是非线性的，在资源受限的 RFID 设备上解决非线性问题使用模糊系统[15,16]可以简化复杂的数学公式计算，并且输出含有多个模糊规则的合并，即使有一个规则失败了，其他规则可以补偿和扩展，解决了系统灵敏度和稳定性的矛盾以及和人类思维的一致性，达到了较高的可靠性，在本节中讨论了一种使用模糊控制系统的方法实现了 RFID sensor 网络的混合多参数的资源配置控制模型，提高了系统的性能。在图 5 中描述了一个 RFID 系统资源配置控制框架。在资源配置框架中，由资源侦听器来监控和捕获资源状态，其中 *ResourceAttributeListener*={*B*,*M*,*C*,*E*}获取 QoS 属性及其数值，作为输入。资源管理器是配置环节中重要的一个器件，其中有 3 个模糊控制器，分别为 $U_{\text{RFID-Tg}}$ 模糊控制器、$U_{\text{RFID-Rd}}$ 模糊控制器和 $U^{\text{Active}}_{\text{RFID-System}}$ 模糊控制器。捕获的资源属性值及资源优化参数误差值输入到这些模糊控制器中，推理产生资源配置 Action 输入到对应的 3 个适配器中，即 $U_{\text{RFID-Tg}}$ 适配器、$U_{\text{RFID-Rd}}$ 适配器和 $U^{\text{Active}}_{\text{RFID-System}}$ 适配器，然后由适配器调用 ConfigurationPerformed 来执行资源调节配置。在上述介绍的 3 种配置模型中，面向 $U_{\text{RFID-Tg}}$ 的控制模型配置 QoS 参数会影响到 RFID tag 到 RFID reader 之间的数据传输的吞吐量。面向

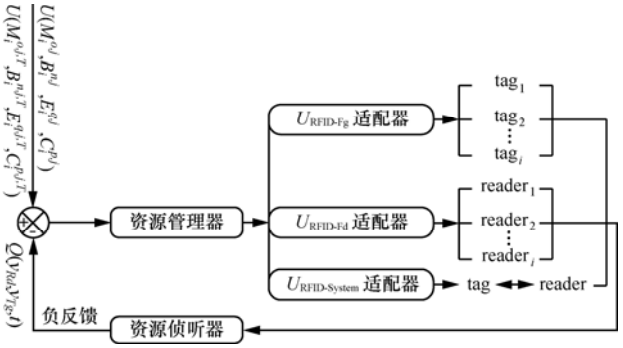

图 5 RFID 系统资源配置控制框架

$U_{\text{RFID-Rd}}$ 的控制模型配置 QoS 参数会影响到 RFID reader 之间数据传输的路由变化。面向 $U_{\text{RFID-Tg}}$ 和 $U_{\text{RFID-Rd}}$ 的控制模型配置 QoS 参数同时也会影响到各自的性能。考虑到这 3 种情况，本文设计了 3 种资源配置策略。

4.1 面向 $U_{\text{RFID-Tg}}$ 和 $U_{\text{RFID-Rd}}$ 的资源配置策略

面向 $U_{\text{RFID-Tg}}$ 的 QoS 参数调整如图 6 所示，是调节 RFID tag 服务 agent 参数的方法。资源侦听器获取 RFID tag 服务 agent 中标签的资源值。资源配置访问适配器，是配置环节中重要的一个器件，通过对资源管理器的调用来更新资源参数，该适配器通过配置文件对 RFID tag 进行资源优化配置。资源侦听器用来监控和捕获资源状态，其中，ResourceAttri- buteListener 获取 QoS 属性及其数值，资源配置访问适配器中的 ConfigurationPerformed 用来从资源管理器中获取需要配置的资源类型。

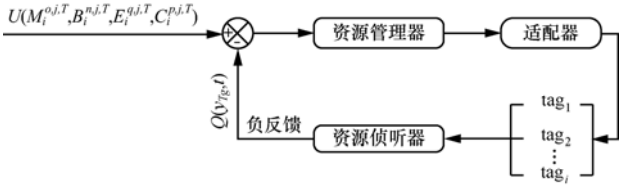

图 6 RFID tag 资源配置控制模型

$U_{\text{RFID-Tg}}$ 型调节通过检测 tag 的 QoS 资源质量，主动反映其资源状况，使 RFID tag 和 reader 之间的数据通信达到最佳平衡。其次，tag 有时存储 sensor 数据，并传输给 reader。由拉格朗日效用函数 $U_{\text{RFID-Tg}}$ 求出的值 $U(M_i^{o,j,T},B_i^{n,j,T},E_i^{q,j,T},C_i^{p,j,T})$ 作为期望的值并且选择性调节各个参数。$U_{\text{RFID-Rd}}$ 型调节如图 7 所示，是通过检测 reader 的 QoS 资源质量，主动反映其资源状况，使 RFID tag 和 reader 之间的数据通信达到最佳平衡。其次，reader 接收、存储、处理 tag/sensor 数据，有时不能满足实际资源需求，由拉格朗日效用函数 $U_{\text{RFID-Rd}}$ 求出的 $U(B_i^{n,j},M_i^{o,j},C_i^{p,j},E_i^{q,j})$ 值作为期望的值选择性调节各个参数。

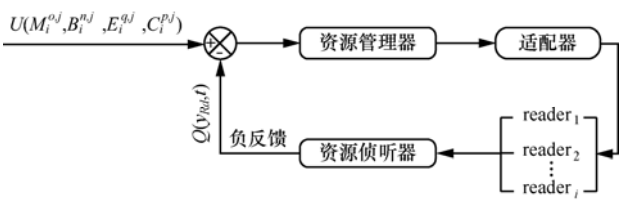

图 7 RFID reader 资源配置控制模型

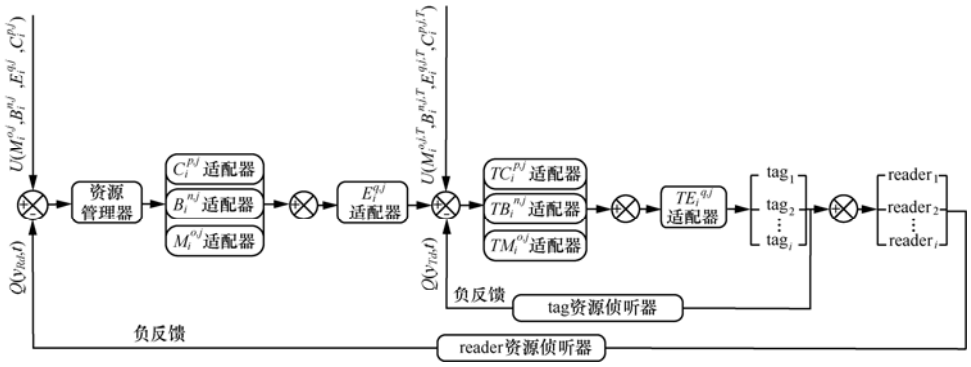

图 8　RFID tag 和 reader 联合资源配置控制模型

4.2　面向 $U_{\text{RFID-System}}$ 的资源配置策略

$U^{\text{Active}}_{\text{RFID-System}}$ 型资源配置模式如图 8 所示，是 tag 和 reader 之间相互调节其资源参数来获得资源的最佳配置，如功率和带宽以及存储和 CPU 处理的联合调节。

在 $U^{\text{Active}}_{\text{RFID-System}}$ 型调节过程中，首先，tag 发送请求需要的带宽容量、数据分组的大小及处理任务规模给 reader 控制器，根据这些参数 reader 主动调节其参数，限定其读取范围，以解除对其他 reader 的干扰，并且决定其对每个 tag 分配的带宽和时间隙，而 tag 根据 reader 的资源配置反馈输出调节其资源配置。

4.3　RFID 系统资源配置算法

4.3.1　资源侦听器算法实现

在上文中提到的 3 种配置策略类型中，都需要资源侦听器来监听 IOT 设备的资源状况，资源侦听器是时间驱动的，在给定的时间间隔[t_1,t_2]内侦听 tag 和 reader 中 e_c、c_g、c_c、m_c 组件的资源状态参量，把这些值送入到资源管理器。具体算法如图 9 所示。

```
Algorithm ResourceListener()
Class GetResourceSateListener
{Resourcelist contain=resourcelist.getInstance(rfid.device(t))
GetCpuResourceSateParameter()
{ return s_Cpu^i(t)}
GetMemoryResourceSateParameter()
{return s_Memory^i(t)}
GetEnergyResourceSateParameter()
{return s_Energy^i(t)}
GetBandwidthResourceSateParameter()
{return s_Bandwidth^i(t)}}
UpdateResourceSate()
{ Ps=CreateInstance(rfid.device)
Gs=new GetResourceSateListener()
Ps.set(S_C,Gs.GetCpuResourceSateParameter())
Ps.set(S_M,Gs.GetMemoryResourceSateParameter())
Ps.set(S_E,Gs.GetEnergyResourceSateParameter())
Ps.set(S_B,Gs.GetBandwidthResourceSateParameter())
}}
```

图 9　资源侦听器算法

4.3.2　资源管理控制器算法实现

RFID tag 和 reader 的 QoS 参数模糊控制器根据 parameterlist 调整策略来改变其值，RFID 系统的 QoS 值与 tag 和 reader 的 QoS 值相关，因此通过改变 tag 和 reader 的 QoS 值就可以改变 RFID 系统的 QoS 值。使用了 Takagi-Sugeno 型模糊推理及 triangle MF 来实现模糊控制，其中模糊控制器以 QoS 误差和误差变化作为输入变量，然后产生配置输出到 tag 和 reader 组件中。根据效用函数求出的值为系统最大效用的最优资源值，作为每个 tag i 期望的 QoS 值，其中包括：Q^i_C、Q^i_M、Q^i_E、Q^i_B。当前值为：S^i_C、S^i_M、S^i_E、S^i_B。误差为

$$e(t)_C = QoS(S^i_C,t) - Q^i_C, e(t)_M = QoS(S^i_M,t) - Q^i_M,$$
$$e(t)_E = QoS(S^i_E,t) - Q^i_E, e(t)_B = QoS(S^i_B,t) - Q^i_B \quad (21)$$

误差集合为：$e(t) = \{e(t)_C, e(t)_M, e(t)_E, e(t)_B\}$

在误差集合中有时只需使用 $e(t)_M$ 来调整数据存储。如存储器中的标签数据超过其上限，要删除其旧的数据。此时其他的误差权重为零。因此，在不同的时刻，根据不同的权重来配置其资源。误差权重表达式为

$$We(t) = w(t)_C e(t)_C + w(t)_M e(t)_M + w(t)_E e(t)_E + w(t)_B e(t)_B \quad (22)$$

误差模糊数集合为

$$L_e = \{L^C_e, L^M_e, L^E_e, L^B_e\}$$
$$L^C_e = \{f^C_1, \cdots, f^C_m\}, L^M_e = \{f^M_1, \cdots, f^M_m\}$$
$$L^E_e = \{f^E_1, \cdots, f^E_m\}, L^B_e = \{f^B_1, \cdots, f^B_m\} \quad (23)$$

误差变化为

$$\Delta e(t)_C = e(t)_C - e(t-1)_C, \Delta e(t)_M = e(t)_M - e(t-1)_M,$$
$$\Delta e(t)_E = e(t)_E - e(t-1)_E, \Delta e(t)_B = e(t)_B - e(t-1)_B \quad (24)$$

误差变化集合为

$$\Delta e(t)=\{\Delta e(t)_C,\Delta e(t)_M,\Delta e(t)_E,\Delta e(t)_B\} \tag{25}$$

对应的误差变化权重为

$$W^{\Delta}\Delta e(t)=w(t)_C^{\Delta}\Delta e(t)_C+w(t)_M^{\Delta}\Delta e(t)_M+ w(t)_E^{\Delta}\Delta e(t)_E+w(t)_B^{\Delta}\Delta e(t)_B \tag{26}$$

误差变化模糊数集合为

$$\begin{aligned}&L_{\Delta e}=\{L_{\Delta e}^C,L_{\Delta e}^M,L_{\Delta e}^E,L_{\Delta e}^B\},\\&L_{\Delta e}^C=\{\Delta f_1^C,\cdots,\Delta f_m^C\},L_{\Delta e}^M=\{\Delta f_1^M,\cdots,\Delta f_m^M\},\\&L_{\Delta e}^E=\{\Delta f_1^E,\cdots,\Delta f_m^E\},L_{\Delta e}^B=\{\Delta f_1^B,\cdots,\Delta f_m^B\}\end{aligned} \tag{27}$$

S_C^i，S_M^i，S_E^i，S_B^i 对应的 4 个模糊调节参数为 P_1^C、P_2^M、P_3^E、P_4^B，模糊规则表示如下：

if $r_e=f_i\wedge\Delta f_i=r_{\Delta e}(f_i\in L_e,\Delta f_i\in L_{\Delta e})$ then

$$\begin{cases}p_1^C=\alpha_{i,1}r_e^C+\beta_{i,1}r_{\Delta e}^C+\gamma_{i,1}\\p_2^M=\alpha_{i,2}r_e^M+\beta_{i,2}r_{\Delta e}^M+\gamma_{i,2}\\p_3^E=\alpha_{i,3}r_e^E+\beta_{i,3}r_{\Delta e}^E+\gamma_{i,3}\\p_4^B=\alpha_{i,4}r_e^B+\beta_{i,4}r_{\Delta e}^B+\gamma_{i,4}\end{cases} \tag{28}$$

$\begin{bmatrix}\alpha_{i,1},\beta_{i,1},\gamma_{i,1}\\\alpha_{i,2},\beta_{i,2},\gamma_{i,2}\\\alpha_{i,3},\beta_{i,3},\gamma_{i,3}\\\alpha_{i,4},\beta_{i,4},\gamma_{i,4}\end{bmatrix}$ 是第 i 个模糊规则增益系数，规则推理单元如图 10 所示。

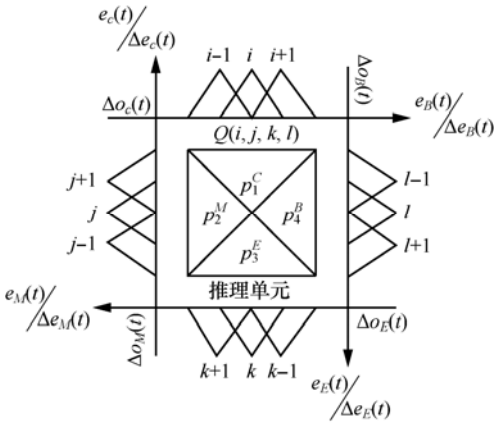

图 10　4 条规则的模糊推理单元

在图 10 中的 Q 点同时触发 4 条规则。反模糊化输出为

$$\begin{cases}\Delta o_C(t)=\dfrac{\sum_{i,j}k_C(i,j)\mu_C(i,j)}{\sum_{i,j}\mu_C(i,j)}\\\Delta o_M(t)=\dfrac{\sum_{i,j}k_M(i,j)\mu_M(i,j)}{\sum_{i,j}\mu_M(i,j)}\\\Delta o_E(t)=\dfrac{\sum_{i,j}k_E(i,j)\mu_E(i,j)}{\sum_{i,j}\mu_E(i,j)}\\\Delta o_B(t)=\dfrac{\sum_{i,j}k_B(i,j)\mu_B(i,j)}{\sum_{i,j}\mu_B(i,j)}\end{cases} \tag{29}$$

模糊调节参数取值范围：

$$\begin{aligned}&U_C^{\mathrm{Min}}\leqslant p_1^C\leqslant U_C^{\mathrm{Max}},U_M^{\mathrm{Min}}\leqslant p_2^M\leqslant U_M^{\mathrm{Max}},\\&U_E^{\mathrm{Min}}\leqslant p_3^E\leqslant U_E^{\mathrm{Max}},U_B^{\mathrm{Min}}\leqslant p_4^B\leqslant U_B^{\mathrm{Max}}\end{aligned} \tag{30}$$

1) CPU/MCU 使用率控制算法

CPU/MCU 使用率的调节主要是避免过多的数据接收、过滤、处理使其过度使用而影响 RFID 系统的性能。对于每一次处理任务定义为 $DT_i(1\leqslant i\leqslant N)$ N 是周期数[17]。$DT_i=(T_i^C,T_i,U_i)$，T_i^C 是估计任务的执行时间。T_i 是处理的周期，且 $T_{i,\mathrm{Min}}\leqslant T_i\leqslant T_{i,\mathrm{Max}}$，$T_{i,\mathrm{Min}}$ 是最小执行周期，$T_{i,\mathrm{Max}}$ 是最大执行周期。U_i 是 CPU/MCU 的执行效用，且 $U_i=\dfrac{T_i^C}{T_i}$，CPU/MCU 总的效用为：$L^C=\sum_{i=1}^{N}U_i$ $(U_C^{\mathrm{Min}}\leqslant L^C\leqslant U_C^{\mathrm{Max}})$，CPU/MCU 控制器通过测量任务 DT_i 的处理周期来调节工作负载。算法如图 11 所示。

Algorithm Configuration CPU/MCUAction()

1. $e_T(t)=T_u(t)-T_i$
2. $\Delta e_T(t)=e_T(t)-e_T(t-1)$
3. $\Delta T(t+1)=F_T\{e_T(t),\Delta e_T(t)\}$
4. $p_1^C=T_u(t)-\Delta T(t+1)$
5. $\Delta w(t+1)=\Gamma(p_1^C)$
6. $\Gamma(p_1^C)=\begin{cases}\Delta w(t+1),\text{if } p_1^C>0\\0\quad\quad\quad,\text{if } p_1^C=0\\-\Delta w(t+1),\text{if } p_1^C<0\end{cases}$
7. $U_C(t+1)=\begin{cases}U_C(t)+\Gamma(p_1^C),\text{if } p_1^C<0\\U_C(t),\text{if } p_1^C=0\\U_C(t)+\Gamma(p_1^C),\text{if } p_1^C>0\end{cases}$
8. $T_i^C(t+1)=\begin{cases}1/\alpha_{i,1}T_i^C(t),\text{if } p_1^C>0\\T_i^C(t),\text{if } p_1^C=0\\\alpha_{i,1}T_i^C(t),\text{if } p_1^C<0\end{cases}$
9. $Send(T_i^C(t+1),U_C(t+1),CPU)$

图 11　CPU/MCU 使用率控制算法

2) 存储控制算法

Memory 调节控制算法主要监测 buffer 和存储器的值，若超过阈值则删除或迁移旧的数据以扩大存储空间，算法如图 12 所示。

Algorithm ConfigurationMemoryAction()

1. $e_M(t)=QoS(S_M^i,t)-Q_M^i$

2. $\Delta e_M(t)=e_M(t)-e_M(t-1)$

3. $\Delta M(t+1)=F_M\{e_M(t),\Delta e_M(t)\}$

4. $p_2^M=QoS(S_M^i,t)-\Delta M(t+1)$

5. $QoS(S_M^i,t+1)=\begin{cases}\alpha_{i,2}QoS(S_M^i,t),\text{if } p_2^M\leqslant U_M^{\min}\\ QoS(S_M^i,t),\text{if } U_M^{\min}\leqslant p_2^M\leqslant U_M^{\max}\\ 1/\alpha_{i,2}\,QoS(S_M^i,t),\text{if } p_2^M\geqslant U_M^{\max}\end{cases}$

6. $Send(QoS(S_M^i,t+1),Memory)$

图 12　存储控制算法

3) 能量控制算法

能量控制算法使用资源侦听器件监控电池能量状态，通过(BatteryLevel)SystemState.GetValue 获取电池实际使用参数 SystemProperty.PowerBattery 来控制能量的输出。在能量的调节决策中包括 3 种调节，若调节参数 $U_E^{\min}\leqslant p_3^E\leqslant U_E^{\max}$ 时,E 不变，若 $p_3^E\geqslant U_E^{\max}$ 时，E 变小，若 $p_3^E<U_E^{\min}$ 时，E 变大，算法如图 13 所示。

Algorithm ComfigurationEnergyAction()

1. $e_E(t)=QoS(S_E^i,t)-Q_E^i$

2. $\Delta e_E(t)=e_E(t)-e_E(t-1)$

3. $\Delta E(t+1)=F_E\{e_E(t),\Delta e_E(t)\}$

4. $p_3^E=QoS(S_E^i,t)-\Delta E(t+1)$

5. $QoS(S_E^i,t+1)=\begin{cases}\alpha_{i,3}QoS(S_E^i,t),\text{if } p_3^E\leqslant U_E^{\min}\\ QoS(S_E^i,t),\text{if } U_E^{\min}\leqslant p_3^E\leqslant U_E^{\max}\\ 1/\alpha_{i,3}\,QoS(S_E^i,t),\text{if } p_3^E\geqslant U_E^{\max}\end{cases}$

6. $Send(QoS(S_E^i,t+1),Energy)$

图 13　能量控制算法

4) 功率和带宽联合调节算法

在图 8 中描述的反馈调节控制系统，对于功率和带宽联合调节以解除 reader 之间的干扰以及解决 tag 和 reader 之间的速率不匹配问题，提高标签数据的读取率。首先，tag 向 reader 申请传输资源，即比特率，reader j 确定为 n 个 tag i 在 l 次传输时提供的 $SIR_j^r(t)$ 为 $r_j^l=\dfrac{R_i^l}{bw_j}\dfrac{E_b}{I_0}$ [18,19]，通过对 tag 的通道发送速率的调节，使得 tag 每次发送速率是可变的，有 l 个不同的速率 R_i^l，在 reader 中的模糊控制器由 resourcelist 获取 $SIR_j^r(t)$ 当前值,计算与 tag 的期望 $SIR_j^t(t)$ 的误差及其变化，tag 的期望 $SIR_j^t(t)$ 下，当获取的 $SIR(t)\geqslant r_j^l$ 时，那么 tag 数据被 reader 成功接收，否则不能被成功接收。可见在功率和带宽联合调节的情况下，传输的成功与否取决于 tag 使用什么样的速率，也即，若调节参数 $p_4^B\geqslant r_j^{l+1}$，说明 reader 通道质量好，可以增大 tag 的传输速率。若 $r_j^l\leqslant p_4^B\leqslant r_j^{l+1}$ 此时 reader 通道质量没有变化，tag 的发送速率保持不变。若 $p_4^B<r_j^l$，此时 reader 的通信质量很差，应该减慢 tag 的发送速率。若 $p_4^B<r_j^{\mathrm{Min}(l)}$，此时,tag 遇到了严重的 reader-to-reader 干扰，导致了无效的传输。为了节约 tag 的能量，这时 reader 应命令 tag 停止发送数据，使得 tag i 在第 t+1 时间没有传输功率,为避免无限期的等待传输，设置延迟时间为 $Q_T(y_{tg},t)_{\mathrm{delay}}$，若 $\tau_{tg}>Q_T(y_{tg},t)_{\mathrm{delay}}$，tag i 此次数据传输结束，等待下一次传输。调节算法如图 14 所示。

Algorithm ConfigurationUnionAdjustPower_BandwidthAction()

1. $r_j^l=\dfrac{R_i^l}{bw_j}\cdot\dfrac{E_b}{I_0}$

2. $e_{PB}^1(t)=SIR_j^r(t)-SIR_j^t(t)$

3. $\Delta e_{PB}^1(t)=e_{PB}^1(t)-e_{PB}^1(t-1)$

4. $\Delta PB_1(t+1)=F_{PB}^1\{e_{PB}^1(t),\Delta e_{PB}^1(t)\}$

5. $p_4^B=SIR_j^r(t)-\Delta PB_1(t+1)$

6. $R_i(t+1)=\begin{cases}R_i^{l+1},\text{if } p_4^B\geqslant r_j^{l+1}\text{ and } R(t)\neq R^l\\ R_i^l\ ,\text{if } r_j^l\leqslant p_4^B\leqslant r_j^{l+1}\\ R_i^{l-1},\text{if } p_4^B<r_i^l\text{ and } R(t)\neq R^{\min}\end{cases}$

7. $SIR_j^r(t+1)=\begin{cases}r_j^{l+1},\text{if } p_{41}^B\geqslant r_j^{l+1}\text{ and } R(t)\neq R^l\\ SIR_j^r(t),\text{if } r^l\leqslant p_{41}^B\leqslant r^{l+1}\\ r^{l-1},\text{if } p_{41}^B<r^l\text{ and } R(t)\neq R^{\min}\end{cases}$

8. $e_{PB}^2(t)=SIR_j^r(t+1)-SIR_j^r(t)$

9. $\Delta e_{PB}^2(t)=e_{PB}^2(t)-e_{PB}^2(t-1)$

10. $\Delta PB_2(t+1)=F_{PB}^2\{e_{PB}^2(t),\Delta e_{PB}^2(t)\}$

11. $P(t+1)=P(t)+\Delta PB_2(t+1),U_{E(P)}^{\min}\leqslant P(t+1)\leqslant U_{E(P)}^{\max}$

12. $Send(R_i(t+1),P(t+1),PB)$

图 14　功率和带宽联合调节算法

资源管理器中的 ConfigurationAction()如图 15 所示。

```
Algorithm ConfigurationAction()
ConfigurationAction()
{While(true)do    { parameterlist=receive.getparameterlist(type,e, Δe )
If (parameterlist!=null) then
{type=getparameterlist.get('type')
If type=='CPU' then
ConfigurationCPU/MCUAction();
Elseif type=='Memory' then
ConfigurationMemoryAction();
Else if type=='Energy' then
ConfigurationEnergyAction();
Else ConfigurationUnionAdjustPower_BandwidthAction()
}}}
```

图 15　资源管理器中的 ConfigurationAction()算法

其中，CofigurationAdapter 为资源适配器，是事件驱动，根据不同的配置事件消息通过函数执行相应的配置，函数为 CofigurationPerformed()，其算法如图 16 所示。

```
Algorithm ConfigurationPerformed()
CofigurationPerformed()
{Performedparameterlist=receive.getparameterlist(QoSValue,type)
If (Performedparameterlist!=null) then
{ Ptype=getparameterlist.get('type')
If (Ptype=='CPU') then
{QoS_Configuration.set(QoS(S_i^C,t+1),currQoS(S_i^C))
QoS_Adapter.execute}
Else if (Ptype=='Memory') then
{QoS_Configuration.set(QoS(S_i^M,t+1),currQoS(S_i^M))
QoS_Adapter.execute}
Else if (Ptype=='Energy')then
{QoS_Configuration.set(QoS(S_i^E,t+1),currQoS(S_i^E))
QoS_Adapter.execute}
Else { Union_PerformedReader()
 Union_PerformedTag()}}
```

图 16　CofigurationPerformed()算法

RFID reader 配置函数如图 17 所示。

```
Algorithm UnionPerformedReader()
UnionPerformedReader()
{QoS_UnionReaderConfiguration.set(channel,curr)
QoS_UnionReaderConfiguration.set(R(t+1),currR)
QoS_UnionReaderConfiguration.set(P(t+1),currP)
QoS_Adapter.execute}
```

图 17　RFID reader 配置函数算法

RFID tag 配置函数如图 18 所示。

```
Algorithm UnionPerformedTag()
UnionPerformedTag()
{QoS_UnionTagConfiguration.set(newinterval,curr)
QoS_Adapter.execute}
```

图 18　RFID tag 配置函数算法

5　实验分析

5.1　验证调节算法有效性

验证调节算法有效性主要包括 5 方面：①CPU/MCU 调节的有效性；②Memory 调节的有效性；③功率和带宽联合调节的有效性；④P_1^C、P_2^M、P_3^B 参数混合调节的有效性；⑤P_1^C、P_2^M、P_3^B、P_4^E 混合参数调节的有效性，其中包含有功率和带宽联合调节。根据误差和误差变化的权重，把实验分为 2 种类型，Type$_1$ 是非干扰固定速率型，在无干扰固定速率的情况下多参数联合调节，即 $P^1{}_{W(t)1}= P_1^C W_1^C(t)+ P_2^M W_2^M(t)+ P_3^B W_3^B(t)$,此种情况下，$W_4^E(t)=0$ 时，验证在 P_1^C、P_2^M、P_3^B 混合参数调节的情况下与 $W_2^M(t)=0$ 时，$P^1{}_{W(t)2}= P_1^C W_1^C(t)+P_3^B W_3^B(t)$型调节的 RFID 系统性能比较分析。Type$_2$ 型是干扰和变速率传输型调节，考虑干扰的情况下多参数联合调节，即 $P^2{}_{W(t)1}=P_1^C W_1^C(t)+P_2^M W_2^M(t)+P_3^B W_3^B(t)+P_4^E W_4^E(t)$, 验证在 P_1^C、P_2^M、P_3^B、P_4^E 混合参数调节的情况下分别与 $P^1{}_{W(t)1}$、$P^1{}_{W(t)2}$ 调节的 RFID 系统性能比较分析。

5.2　性能指标

1) 吞吐量的具体衡量指标为平均读取成功率(ARSR, average read successfully rate)

$$ARSR=\frac{\sum_{l=1}^{N_{\text{tag}}} R_l}{S_p} \tag{31}$$

其中，N_{tag} 是在仿真时成功读取标签的数量。R_l 是 tag 在第 l 次成功传输时的速率。对于 tag 的一次不成功传输其传输率为 0，S_p 是采样的周期。

2) 不成功读取概率(PUR, probability of unsuccessfully read)

$$PUR=\frac{l_{rf}}{l_s-l_{tf}} \tag{32}$$

其中，l_{rf} 是在一个仿真中不能成功读取的标签数。l_{tf} 是不能成功获得传输率的标签个数。l_s 是总的标签数。

3) 内存溢出概率(PMO, probability of memory

overflow)

$$PMO = \frac{M_{of}}{M_{tg} - M_{of}} \tag{33}$$

其中，M_{of}是内存溢出的标签个数。M_{tg}是总标签个数。

对 Type$_1$ 型中 $P^1{}_{W(t)1}$是在不考虑干扰的情况下，$P_1{}^C$、$P_2{}^M$、$P_3{}^B$混合参数调节，其实验过程为：创建 CPU、带宽、内存监测线程、reader 接收线程和 tag 发送线程。CPU、带宽、内存监测线程来调节各个参数，观察 reader 在读取过程中的参数调节带来 RFID 系统性能的变化，实验首先启动 tag 发送线程，然后是 reader 接收线程，随着 tag 发送线程发送 tag 数目增加，read 接收线程监测到 tag 数目变化率增大，当数目 $N_{tag} > \rho'$时，reader 上的带宽监测线程创建新的 channel，增加接收通道个数，使数据读取量增加，CPU 监测线程调节加快过滤处理时间周期，内存监测线程监控内存空间的变化，当 $N_M > \rho''$，启动调节线程，转移内存的电子标签数据，接收新的数据。在 $P^1{}_{W(t)2}$型中，测试 $P_1{}^C$、$P_3{}^B$混合参数调节，随着 tag 发送线程发送的 tag 数目的增加，当数目 $N_{tag} > \rho'$时，reader 上的带宽监测线程创建新的 channel，增加接收通道个数，使数据读取量增加，CPU 监测线程调节加快过滤处理时间周期，但在 $P^1{}_{W(t)2}$型中没有内存调节过程，增加了内存溢出的概率导致标签的读取概率降低。在图 19 和图 20 中比较分析了 $P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$中 tag 读取率。设 tag 长度为 16byte，传输速率为 1.6byte/ms 内存大小 M_{Size}=1KB，S_p=1，附着在车辆上的电子标签的移动速度分布为[8,80]km/h。

图 19 为 $P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节平均读取成功率，$P^1{}_{W(t)1}$型调节由于使用本文提出的 $P_1{}^C$、$P_2{}^M$、$P_3{}^B$混合型调节，使得读取成功率稳定上升，而 $P^1{}_{W(t)2}$型调节由于使用 $P_1{}^C$、$P_3{}^B$混合参数调节，没有使用 $P_2{}^M$调节参数，在内存资源被占用的情况下无法调节而使得 tag 的标签数为 70 个时，其读取率达到恒定的水平，这是由于当标签个数为 70 时，此时的内存占用为 70×16=1120byte>1KB 的内存空间，$P_1{}^C$调节参数在不断加快 CPU 的处理，使内存中积累了大量 tag 数据，这时没有 $P_2{}^M$参数调节已经充满内存，reader 内存中的数据无法转移造成了内存溢出概率上升，其次，对于内存中旧的临时数据也没有相应调节来清除，使得读取成功率下降。

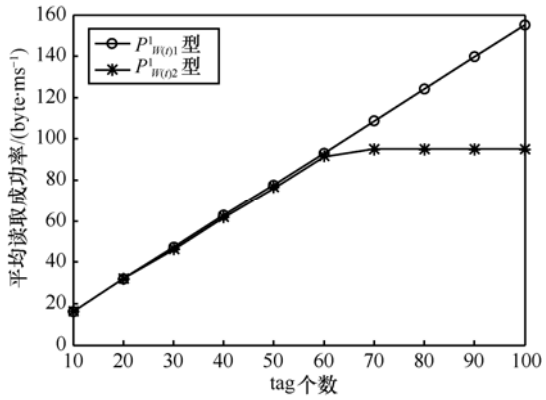

图 19　$P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节平均读取成功率

图 20 为 $P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节不成功读取概率，$P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节在标签个数小于等于 20 时不成功读取概率为 0，随着标签个数的增加，$P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节的不成功读取概率都开始增长，但 $P^1{}_{W(t)1}$型调节的增长保持在平稳的状态，而 $P^1{}_{W(t)2}$型调节却保持快速增长，在标签数为 70 时，其不成功读取概率增长变快，开始加速增长，这是由于没有考虑 $P_2{}^M$参数调节的缘故。

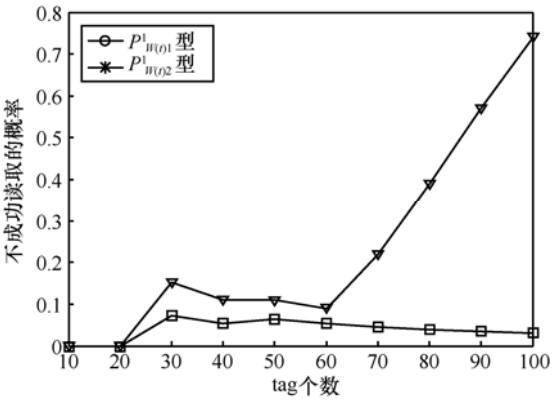

图 20　$P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节不成功读取概率

$P^2{}_{W(t)1}$型调节的实验过程与 $P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节相类似，主要不同在于，在 reader 上创建功率调节线程，检测到其通道状况。在 tag 要创建速率调节线程，当 $p_4{}^B < r_j{}^{\mathrm{Min}(l)}$发生时，tag 遇到了严重的 reader-to-reader 干扰，导致了无效的传输，为了节约 tag 的能量，这时 reader 命令 tag 停止发送数据，而 reader 能量调节参数调节输出功率来解除干扰，tag 在等待一个 $Q_T(y_{tg},t)_{delay}$时间后，重新传输，但当调节参数 $p_4{}^B \geqslant r_j{}^{l+1}$，说明 reader 通道质量好，可以增大 tag 的传输速率，不像 $P^1{}_{W(t)1}$和 $P^1{}_{W(t)2}$型调节是固定速率，致使好的信道质量没法最大利用，reader 带宽资源被浪费。$P^2{}_{W(t)1}$型调节可以使 tag 变

速率传输，即 reader 输出控制命令，tag 接收线程收到命令后，交由速率调节线程调节速率。发送速率分别为：0.16byte/ms、1.6byte/ms、16byte/ms。

图 21 和图 22 中 $P^2_{W(t)1}$ 型调节使用 P_1^C、P_2^M、P_3^B、P_4^E 参数混合调节。P_3^B、P_4^E 参数作用于 *UnionPowerBandwidth* 调节，使得 tag 变速传输，而且 reader 调节输出功率解除 reader-to-reader 干扰，极大地提高了 tag 的平均读取成功率，从而减少了读取不成功概率。在图 21 中当标签数为 90 时，平均读取成功率情况为：$P^2_{W(t)1}$> $P^1_{W(t)1}$> $P^1_{W(t)2}$，$P^2_{W(t)1}$ 型调节保持较高的平均读取成功率。而在图 22 中当标签数为 80 时，读取不成功概率情况为 $P^1_{W(t)2}$> $P^1_{W(t)1}$>$P^2_{W(t)1}$，即 $P^2_{W(t)1}$ 型调节读取不成功概率为最小且保持稳定状态。

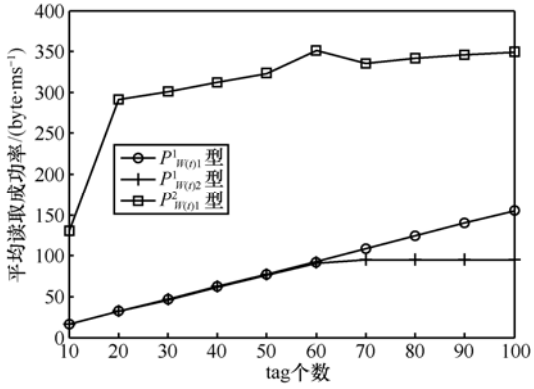

图 21　$P^1_{W(t)1}$、$P^1_{W(t)2}$ 和 $P^2_{W(t)1}$ 型调节平均读取成功率

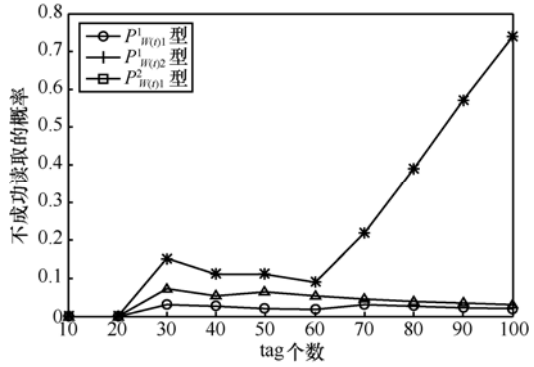

图 22　$P^1_{W(t)1}$、$P^1_{W(t)2}$ 和 $P^2_{W(t)1}$ 型调节读取不成功概率

6　结束语

本文以 RFID sensor 节点的 QoS 组件模型为基础，建立面向服务的 RFID 系统资源消耗模型，使用拉格朗日优化方法求出了各个 QoS 组件资源的调节参数的下界，然后，使用下界作为模糊控制的期望参数来调节资源变量以达到最优资源配置。随后，建立了 3 种资源配置控制框架，分析了 3 种类型资源配置服务策略，并给出了相应的算法，通过这些资源配置服务策略有效防止内存的溢出,节约了能量,调节了 CPU/MCU 的工作负载，使得 tag 和 reader 之间传输速率相匹配以及 reader-to-reader 之间的干扰解除，为物联网下一代智能电子标签和读写器的优化设计提供了理论基础。

参考文献：

[1] ATZORI L, IERA A, MORABITO G. The internet of things: a survey[J]. Computer Networks, 2010, 54(15):2787-2805.

[2] CONTI J P. The internet of things[J]. IET Communications Engineer, 2006, 4(6):20-25.

[3] KRANZ M, HOLLEIS P, SCHMIDT A. Embedded interaction: interacting with the internet of things[J]. IEEE Internet Computing, 2010, 14(2):46-53.

[4] GUINARD D, TRIFA V, KARNOUSKOS S. Interacting with the SOA-based internet of things: discovery, query, selection, and on demand prov-isioning of Web services[J]. IEEE Transactions on Services Computing, 2010, 3(3):223-235.

[5] GUAN Q, LIU Y, YANG Y P. Genetic approach for network planning in the RFID systems[A]. Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications[C]. Los Alamitos, CA, USA, IEEE Computer Society, 2006. 567-572.

[6] CHEN H N, ZHU Y L, HU K Y. Multi-colony bacteria foraging optimizati-on with cell-to-cell communication for RFID network planning[J]. Applied Soft Computing, 2010, 10(2):539-547.

[7] MUSKENS J, CHAUDRON M. Prediction of run-time resource consumption in multi-task component-based software systems[A]. Proceedings of the 7th International Symposium on Component-Based Software Engineering(CBSE'2004)[C]. Edinburgh, UK, Springer, 2004.162-177.

[8] HAMLET D, MASON D, WOITM D. Theory of software reliability based on components[A]. Proceedings of the 23rd International Conference on Software Engineering(ICSE'2001)[C]. Toronto, Ontario, Canada, IEEE Computer Society, 2001.361-370.

[9] YEN I L, KHAN L, PRABHAKARAN B. An on-line repository for embedded software[A]. Proceedings of the 13th IEEE International Conference on Tools with Artificial Intelligence[C]. Dallas, TX, USA, IEEE Compu-ter Society, 2001.314-321.

[10] COOPER K, ZHOU J, MA H. Code parameterization for satisfaction of QoS requirements in embedded software[A]. Proceedings of the Inter-national Conference on Engineering of Reconfigurable Systems

and Algorithms[C]. Las Vegas, Nevada, USA, 2003.58-64.

[11] LI C L, LI L Y. Context aware service provisioning in mobile grid[J]. Journal of Network and Computer Applications, 2011, 34(2): 774-782.

[12] LI B, NAHRSTEDT K. A control-based middleware framework for quality-of-service adaptations[J]. IEEE Journal on Selected Areas in Comm-unications, 1999, 17(9):1632-1650.

[13] HUANG L, KUMAR S, KUO C C J. Adaptive resource allocation for multi-media QoS management in wireless networks[J]. IEEE Transactions on Vehicular Technology, 2004, 53(2):547-558.

[14] MERCADO A, LIU K J R. Adaptive QoS for wireless multimedia networks using power control and smart antennas[J].IEEE Transactions on Vehicular Technology, 2002, 51 (5):1223-1233.

[15] ZHANG Y, ZHANG S, HAN S. Adaptive service configuration approach for quality of service management in ubiquitous computing Envir-onments[J]. Journal of Zhejiang University Science A, 2009, 10(7):964-957.

[16] TSAY D L, CHUNG H Y, LEE C J. The adaptive control of non-linear sy-stems using the sugeno-type of fuzzy logic[J]. IEEE Transactions Fuzzy System, 1999, 7(2):225-229.

[17] BASARAN C, SUZER M H, KANG K. Robust fuzzy CPU utilization control for dynamic workloads[J]. Journal of Systems and Software, 2010, 83(7):1192-1204.

[18] SUNG C W, WONG W S. Power control for multirate multimedia CDMA systems[A]. Proceedings of Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'1999)[C]. New York, USA, 1999. 957-964.

[19] CHEN Y, LIN Y, WANG J. Adaptive fuzzy-based rate management and power control in multimedia CDMA cellular systems[J]. Computer Communications, 2008, 31(10):1901-1910.

作者简介：

刘建华（1978-），男，山西运城人，上海大学博士生，主要研究方向为普适计算、无线传感网络、嵌入式系统。

童维勤（1964-），男，上海人，上海大学教授、博士生导师，主要研究方向为并行程序设计方法、高性能计算技术和应用、嵌入式技术与分布式系统等。