上下文不一致性检测及消除的研究

张奕男 吴 刚
(上海交通大学软件学院 上海 200240)

摘 要 在普适计算环境下,由于获得的上下文信息存在不一致性,使得上下文感知应用无法正常进行响应。研究了普适计算环境下的上下文不一致性的检测及消除,针对其提出了基于置信度数据库的反馈机制,并且丰富和扩展了上下文不一致性的消除算法。实现了相关算法,并通过实验给出了各个算法的效率和正确性的比较。

关键词 上下文不一致性,检测,消除,置信度数据库
中图法分类号 TP311 **文献标识码** A

Research on Context Inconsistency Detection and Resolution

ZHANG Yi-nan WU Gang
(School of Software, Shanghai Jiaotong University, Shanghai 200240, China)

Abstract Abstract The context-aware system requires detection and resolution of context inconsistency, which exists in pervasive computing. Detection and resolution of context inconsistency was studied, a feed-back mechanism based on confidence measure database was presented, and the elimination resolution was enriched and extended in these ways. Also, we accomplished these resolutions, and their performances were evaluated through a series of experiments.

Keywords Inconsistency of context, Detect, Eliminate, Confidence measure database

1 引言

上下文感知是普适计算中的研究热点,而应用从信息源获得的上下文可能存在矛盾,这就使得上下文不一致性的研究变得非常重要。

所谓上下文不一致是指系统中的上下文存在矛盾,它出现的主要原因是上下文感知系统常常假设传感器采集的数据是理想的、精确的,推理规则是绝对合理并且全面的,但这些条件在实际中往往不能满足[1]。具体原因多种多样,文献[2]指出了4种不一致上下文产生的原因:高动态的环境使得上下文容易被废弃;上下文信息可能是由不同标准下的异构来源提供的;推理可能由于计算资源的限制而获得不准确的信息;网络连接的断开或错误而导致不完整的上下文。例如,通过不同的信息来源,得到了张三正在客厅看电视和张三正在厨房吃饭的两条上下文信息,就明显存在矛盾。

对于上下文不一致性的检测及消除,Xu和Cheung提出了基于语义的上下文信息的检测及消除的建模方式[3]并在文献[4]中进行了增量化不一致性检测的研究;而郑笛和朱珊基于这种高级语义上下文信息的检测及消除提出了4种不一致性消除算法[1];文献[5-8]也都对不一致性问题做了一定的研究,但是研究思路和建模形式与文献[1-4]有很大区别。本文采用了文献[3]所提出的建模方式,并据此对上下文不一致性的检测及消除进行了深入的研究。

本文第2节介绍了上下文不一致性研究的建模方式;第3节对上下文不一致性消除算法进行了研究和扩展,并提出了针对上下文不一致性检测及消除的基于置信度数据库的反馈机制;第4节给出了各个算法的效率和正确性的比较结果;最后对本文的工作进行了总结并指出了后续工作。

2 问题建模

本文是针对逻辑性的高级语义上下文进行的不一致性研究。主体建模思想沿用了文献[3]中元组的方式对上下文信息进行抽象和建模。其中主要包括以下概念。

(1)上下文建模

上下文(Context)是由一个七元组表示的:

context = (subject, predicate, object, time, area, certainty, freshness) (1)

式中,上下文信息的主体由一组主谓宾结构进行描述,如"张三,进入,客厅"即对应于(subject, predicate, object)。后面为其他必备条件:时间,地点,确定性,时间戳。

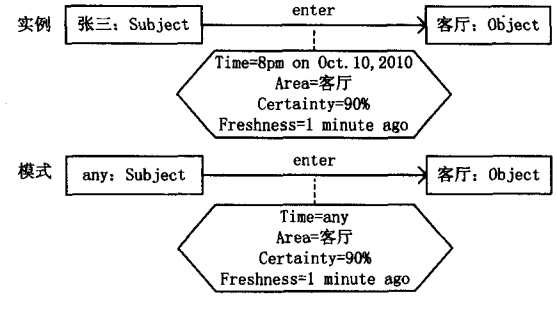

图1 上下文实例和上下文模式

到稿日期:2010-10-20 返修日期:2011-02-18 本文受国家863项目(2009AA01Z123)资助。
张奕男(1986-),男,硕士生,主要研究方向为上下文感知计算、上下文不一致性等,E-mail:zyn8788@163.com;吴 刚(1973-),男,博士,副教授,主要研究方向为情境感知计算、分布计算、普适计算等。

上下文实例(Instant)是指由上面所述的建模方式进行描述的上下文信息,如图 1 所示。

上下文模式(Pattern)是根据上下文实例,对其特定域进行抽象概括得到的能够表征一组上下文实例的抽象模式,如图 1 所示。

(2)约束建模

约束(Constrain)是指在相关的情景下,上下文信息之间存在一定的联系,这些联系被定义为 Constrain,针对特定情景中的所有上下文信息的一系列规则的集合称为组的约束(Constrains)。其中,约束也是用元组的形式表示的:

$$constrain = (rule, j, filed_j, k, filed_k) \quad (2)$$

其中,rule 主要指 5 个语义关系:相等,从属,包含,相交,不相交;对于时间主要包括:早于,晚于,在…期间等关系。而后面 j,k 表示对应的上下文模式的序号,$filed_j$ 和 $filed_k$ 则声明了该规则描述的此模式的特定区域。

3 上下文不一致性的检测及消除

上下文不一致性的检测及消除,依次包括了上下文模式匹配、上下文不一致性检测以及上下文不一致性的消除 3 个主要部分,如图 2 所示。

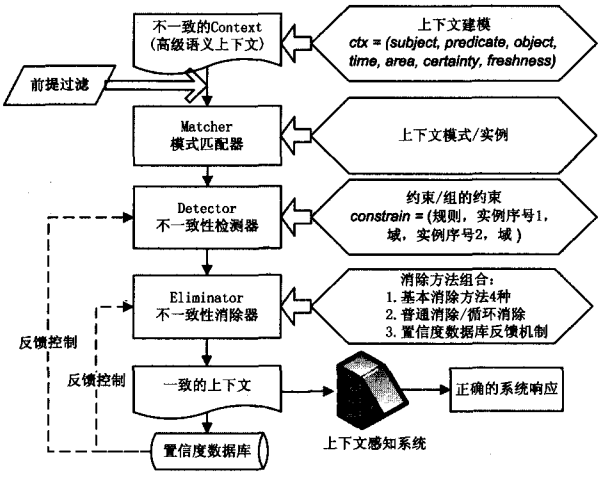

图 2 上下文不一致性检测及消除流程图

其中,上下文模式匹配进行上下文实例到上下文模式的匹配,即寻找满足上下文模式的所有上下文实例;上下文不一致性检测过程中,检测器一旦检测到一组上下文实例满足了某个组的约束条件,就将上下文模式匹配到的结果带入该上下文模式之间的组的约束,对上下文实例进行约束的匹配,通过该不一致性检测后,得到了一组不一致上下文实例对,即一系列的上下文实例间的矛盾关系;上下文不一致性消除的主体思想为,根据不同的消除规则对该检测结果中的不一致的上下文实例进行排序,以规则要求的顺序依次进行清除,直到检测器检测到的上下文实例对都被删除。

下文重点研究了上下文不一致性检测的消除方法:在3.1 节中提出了 4 种上下文不一致性消除的基本算法;3.2 节中提出了循环排序的上下文不一致性消除的扩展算法;3.3 节中提出了针对上下文检测及消除的基于置信度数据库的反馈机制。

3.1 上下文不一致性消除的基本算法

(1) 全丢弃不一致性消除算法:删除所有检测到的不一致的上下文实例。

全丢弃的不一致消除算法利弊分明。优势是实现简单,算法执行效率高。缺点是将有大量正确的上下文信息被一同删掉。

(2) 基于不一致次数的不一致性消除算法:对上下文不一致性检测得到的结果中的所有上下文实例根据其出现的次数进行排序,取出现次数最多的为先。然后依次删除序列中的上下文实例。该算法的关键是上下文信息的错误次数,通过上下文检测而得到不一致上下文实例对,每个实例在上下文实例对中出现一次就表示该实例和其他上下文实例发生了矛盾,而发生矛盾次数较多的,本文将其认定为不一致上下文信息,将其删除,如算法 1 所示。

算法 1 基于不一致次数的不一致性消除算法

1. for 所有匹配到 Patterns 的 Instants:
2. if $instant_i$ 和 $instant_j$ 不满足二者所对应的 Patterns 之间的 constrain:
3. 将($instant_i$, $instant_j$)作为一个不一致上下文实例对存到检测结果 Result_D 中
4. for 所有 Result_D 中的不一致上下文实例对中的每个 Instant:
5. 将每个 Instant 的出现次数进行统计,得到按出现次数排序的不一致 Instants 序列 C
6. while True:
7. 按照顺序依次消除 C 中的 $instant_k$,并消除 Result_D 中的关于 $instant_k$ 的实例对
8. if Result_D 中不再存在不一致的上下文实例对:
9. break,余下的 Instants 即为一致的上下文实例

基于不一致次数的不一致消除算法,由于其对上下文不一致性检测结果的排序等操作使得上下文消除算法的执行时间耗费相对较大。但根据其上下文实例出错次数进行优先删除,使得算法的正确性有了很大的提高,也能为上下文感知应用提供丰富的上下文信息。

(3) 基于不一致出现率的不一致性消除算法:对上下文不一致性检测得到的结果中的所有上下文实例根据其不一致出现率(rate)进行排序,取不一致出现率最高的为先。同理按照排序进行删除。该算法的主旨是依据上下文信息的不一致出现率进行不一致上下文实例排序。

$$rate = \frac{\text{上下文实例出错次数}}{\text{上下文实例被带入上下文模式约束内进行检测的次数}} \quad (3)$$

其中,“上下文实例出错次数”是指上下实例在被检测到的不一致上下文实例对中出现的次数;“上下文实例被带入上下文模式约束内进行检测的次数”是指原始的被带入到上下文不一致性检测及消除算法中的上下文实例的出现次数。而二者的比值表示上下文不一致性在检测器中被检测出的比例,即 rate。

基于不一致出现率的不一致性消除算法,同基于不一致次数的消除算法,有较强的可信性,能够有效地清除上下文的不一致。但是算法同样需要进行排序等操作,时间耗费较大。

(4) 基于相关性的不一致性消除算法:对上下文不一致性检测得到的结果中的所有上下文实例的相关性进行排序,取关联次数最多的为先。其中,本文以上下文实例带入相对应的上下文模式所满足的组的约束而进行上下文不一致性检测的次数来统计上下文实例的相关性。

基于相关性的不一致性消除算法,由于进行了相关性的

排序，使得时间耗费上也较大。并且，由于该相关性仅仅表征了上下文实例对应的上下文模式之间相关联的程度，与上下文不一致的矛盾本身并没有绝对的关系，因此经过该算法消除后的上下文实例的正确性不高。故而该算法可以作为其他算法的补充和扩展。

上述4种上下文不一致性消除算法，与文献[1]中提出的上下文不一致性消除算法的主要区别是：文献[1]中不一致性的检测是以触发器的形式实现的，一旦检测到一个矛盾的上下文实例对，就根据消除算法对其进行分析并消除；而本文的不一致性消除算法则是建立在一系列上下文实例进行不一致性检测而得到一系列上下文实例之间的不一致关系上的，并由此提出了4种消除算法以及接下来3.2节中介绍的循环排序的不一致性消除算法的扩展。

3.2 上下文不一致性消除的循环排序算法扩展

本节在上下文不一致性消除的基本算法的基础上，根据3.1节中提出的第2,3,4种以不一致性检测结果的排序为消除依据的算法，提出了一种循环排序的方式对其进行扩展。

如图3所示，循环排序消除算法是在普通排序消除算法的基础上，增加了循环排序演变而来的。从上下文不一致性检测得到的一系列不一致的上下文实例对，继而得到根据消除算法进行排序后的实例序列。因为在消除不一致的过程中，当消除最优先删除的不一致上下文实例后，其他的相关的上下文实例在整个上下文实例组内的关联程度和不一致性排序也都发生了变化，所以，为了更加准确地获得各实例在当前实例组内的不一致排序，在每次清除掉最优先不一致实例后，都根据同一排序算法对余下所有不一致实例对内的上下文实例重新排序。并循环排序清除最优先选项，直到清除上下文实例组中所有的不一致上下文。

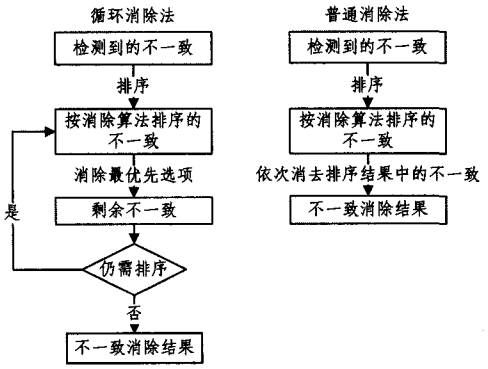

图3 普通消除算法和循环排序消除算法

该循环排序算法是对上下文消除算法的有效扩展，能使最终获得的上下文信息的准确性更高。但是由于循环排序的引入，其消除算法的时间消耗将进一步增大。

3.3 基于置信度数据库的反馈机制

置信度是指评估某要素可靠性的指标，通常用百分数概率表示[9]。

本文中的置信度用来表征上下文模式的可靠性。例如上下文模式“某人进入了浴室”，表示浴室内传感器等感应设备协同工作并进行推理后得到的上下文信息。当多次获得了与该上下文模式所匹配的上下文实例信息并进行了上下文一致性检测及消除后，该上下文模式内所匹配的上下文实例出现的次数和被不一致消除器所清除的次数的比值将有效表征符合该模式的上下文实例的出错概率。简单来说，这里的置信度即为$\frac{\text{正确数}}{\text{总数}}$。

据此，本文提出了针对上下文不一致性检测及消除的基于置信度数据库的反馈机制，如图2所示。其主体思想为：将上下文不一致性的检测及消除的结果以正确率的形式录入数据库，并通过阈值过滤的方式来对上下文不一致检测器和消除器进行反馈控制。首先需要设定正确率的阈值，再利用该阈值对置信度进行评估，据此对于上下文匹配器匹配到的结果或者对上下文不一致性检测器检测到的结果进行筛选，得到精简的上下文不一致性检测器或者消除器的输入。从而大幅度降低上下文不一致性检测及消除算法的时间消耗。

4 性能测试

实验是在一台Windows 7操作系统，4G DDR3内存，以及2.9GHz AMD Athlon(tm)ⅡX4 635 CPU的主机上进行的。

4.1 上下文不一致性检测及消除算法性能分析

本小节对上下文不一致性检测及消除算法的性能进行了分析，着重于采用4种普通形式以及3种循环消除算法后的正确性和时间消耗的比较。

对每次的上下文模式匹配以及检测及消除进行100次循环执行而得到的。通过表1中数据趋势的分析，可以推断出各个算法的优劣。

表1 上下文不一致性检测及消除算法及循环消除算法结果

实例总数（不一致）	全丢弃（时间s/正确数）	基于次数（时间s/正确数）	次数循环（时间s/正确数）	基于出现率（时间s/正确数）	出现率循环（时间s/正确数）	基于相关性（时间s/正确数）	相关性循环（时间s/正确数）
6(2)	0.207/3	0.295/5	0.321/6	0.289/3	0.318/3	0.242/2	0.252/2
7(1)	0.314/4	0.396/7	0.397/7	0.406/7	0.409/7	0.333/3	0.354/3
7(2)	0.313/4	0.403/7	0.409/7	0.413/7	0.416/7	0.341/3	0.352/3
7(2)	0.251/4	0.391/5	0.410/7	0.391/5	0.422/7	0.293/3	0.313/3
8(2)	0.324/5	0.456/6	0.453/8	0.474/6	0.486/6	0.367/4	0.387/4
8(3)	0.297/5	0.406/6	0.412/6	0.483/6	0.514/6	0.414/6	0.422/6
9(2)	0.364/5	0.507/9	0.542/9	0.535/9	0.536/9	0.464/7	0.484/6
9(3)	0.371/6	0.513/6	0.519/6	0.522/6	0.519/6	0.506/6	0.513/6
10(2)	0.500/7	0.604/10	0.635/10	0.551/8	0.608/10	0.531/7	0.554/6
10(3)	0.472/7	0.574/8	0.589/8	0.596/6	0.590/6	0.536/5	0.542/5
10(3)	0.417/6	0.549/8	0.567/8	0.580/6	0.569/6	0.526/5	0.533/5
10(4)	0.443/6	0.534/7	0.581/10	0.505/7	0.607/10	0.472/5	0.490/5
总数101(29)	4.273/62	5.628/84	5.835/92	5.745/76	5.994/83	5.025/56	5.196/54

(a) 全消除算法的时间消耗最少，但正确率无法得到保证，并且由于删除的上下文数量过多，因此只能作为参考方法。基于相关性的消除算法的正确率也不理想，但其独立的排序方式使其成为其他消除算法的有效补充。

(b) 基于不一致次数及其循环消除算法的时间消耗较全丢弃算法和基于相关性算法更大，但其正确率有了很大的保证，并且其在每组实例的检测消除情况上也较为稳定，为最优算法。而基于不一致性出现率及循环算法，受限于实例规模的影响，准确率稍低，但其在时间消耗上和基于不一致性次数的算法不相伯仲，使其可以成为次优算法。

(c) 以(b)中所述最优和次优算法为准，循环算法的正确率较普通消除算法的正确率高出近10%，但在时间消耗上则增加了近5%。所以，在时间消耗允许的范围内，循环消除算法的扩展很好地提高了不一致性检测及消除算法的可靠性。

(下转第129页)

Int J CompCognit,2004,2(2):81 - 130.
[5] 赵耀,王红星,袁保宗. 分形图像编码研究的进展[J]. 电子学报,2000,28(4):95-101,106
[6] Liu M Q,Zhao Y,et al. A fast fractal image coding algorithm based on FGSE[A] // Signal Processing[C]. Beijing: IEEE Press,2006:1153-1156
[7] Bobtsov A,Nikolaev N,Slita O. Control of Chaotic Oscillations of a Satellite[J]. Applied Mathematics and Mechanics(English Edition),2007,28(7):893-900
[8] 屈文建,薛锦云. PAR 方法和循环不变式的范畴语义[J]. 计算机工程与应用,2009,45(8):50-54
[9] Xue J Y. A unified approach for developing efficient algorithmic programs[J]. Journal of Computer Science and Technology,1997,12(4):314-329
[10] Xue JY. Two new strategies for developing loop invariants and their applications[J]. Journal of Computer Science and Technology,1993,8(2):147-154
[11] Dijkstra E W. A discipline of programming [M]. Englewood Cliffs:Prentice Hall,1976
[12] 高军,孙博玲. KOCH 曲线的 DELPHI 程序设计[J]. 电脑学习,2000(2):35-36
[13] 孙继军,卢玉蓉. Von Koch 曲线的 Visual C++程序实现[J]. 攀枝花学院学报,2004,21(1):90-92

(上接第 118 页)

4.2 基于置信度数据库的反馈机制性能分析

本节针对基于置信度数据库的反馈机制对上下文不一致性检测及消除的时间消耗的影响进行实验研究。实验使用的上下文不一致性消除算法为基于上下文不一致性次数的循环消除算法;并且为了便于进行置信度调整,将数据库内存储的关于每个上下文模式相匹配的上下文实例带入进行检测消除的总实例数设为 100;调整其中成功和失败的实例数,即改变了该上下文模式的置信度,从而可以实现对检测器和消除器的不同的反馈机制,使其能够对二者分别产生作用。实验结果如图 4 所示。

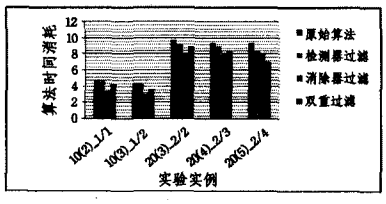

图 4 基于置信度数据库的反馈机制对上下文不一致性检测及消除算法的影响

其中,纵坐标表示时间消耗;横坐标表示带入实验的实例组,具体表示为"实例数(不一致数)_将被检测器过滤实例数/将被消除器过滤实例数"。实验所得算法时间为单次上下文不一致性检测及消除循环执行 1000 次并重复运行 5 次取其平均值所得到。本文将检测器过滤的阈值预设为 0.1,即上下文模式的不一致性置信度低于 0.1 将被认定为错误的上下文实例,不进行不一致性检测即将其直接消除。而消除器过滤的阈值预设为 0.2,同理,一旦该上下文模式被检测到和其他上下文模式存在不一致,即将其直接消除,不执行上下文不一致消除算法。

如图 4 所示,采用了基于置信度数据库的反馈机制的上下文检测及消除算法在时间耗费上明显少于原始的算法。这其中,检测器过滤的力度较小,所以降幅不明显,说明该反馈对于检测器和原始算法的影响较小。但最后一组实验实例说明,当实例设计及规模不定的情况下,检测器过滤也有一定的效果。消除器过滤,虽然只在消除器处进行了算法优化,但由于消除算法在整体算法中的主导地位以及阈值的设定可以过滤掉较多的不一致上下文实例,使得该反馈机制的时间耗费降幅较为明显。而双重过滤的方式在算法的时间消耗上较少,但是需要综合考虑数据库查询的次数,所以对其的使用需要根据不同情况进行分析。

结束语 本文以上下文不一致性的检测及消除算法的研究为核心,提出了 4 种上下文不一致性消除算法和循环排序的消除算法扩展,同时提出了针对上下文检测及消除的基于置信度数据库的反馈机制,并对其正确性和时间消耗进行了评估,丰富了上下文不一致性检测及消除的框架和方法。后续工作将从以下几方面入手:针对置信度数据库在阈值方面的设定对上下文不一致性检测及消除算法的影响进行研究;对上下文不一致性检测及消除流程图所示的前提过滤条件进行研究,希望能有效地过滤上下文实例的输入从而提高上下文不一致性的检测及消除的效率;针对不同领域的上下文信息进行评估,研究本文提出的算法在不同领域模型中的实际应用情况。

参 考 文 献

[1] 郑笛,朱珊. 普适计算环境下上下文不一致性的消除算法研究[J]. 计算机应用研究,2009,26(1):152
[2] 郑迪. 基于上下文感知服务的构件化中间件关键技术研究[D]. 长沙:国防科学技术大学,2008
[3] Xu C,Cheung S C. Inconsistency detection and resolution for context-aware middleware support[C]// the Joint 10th European Software Engineering Conference and 13th ACM SIGSOFT Symposium on the Foundations of Software Engineering(ESEC/FSE 2005). Lisbon:ISBN,2005:336-345
[4] Xu C,Cheung S C,Chan W K. Incremental consistency checking for pervasive context[C]// the 28th International Conference on Software Engineering(ICSE 2006). Shanghai:ISBN,2006:292-301
[5] Huang Yu,Yu Jian-ping,Cao Jian-nong,et al. Checking Behavioral Consistency Constraints for Pervasive Context in Asynchronous Environments[D]. Nanjing:Nanjing University,2009
[6] Huang Yu,Yu Jian-ping,Cao Jian-nong,et al. Concurrent Event Detection for Asynchronous Consistency Checking of Pervasive Context[D]. Nanjing:Nanjing University,2009
[7] Huang Yu,Yu Jian-ping,Cao Jian-nong,et al. A Probabilistic Approach to Consistency Checking for Pervasive Context[D]. Nanjing:Nanjing University,2008
[8] 叶光昶. 普适计算环境下自适应技术研究[D]. 上海:上海交通大学,2007
[9] 百度百科,置信度[OL]. http://baike.baidu.com/view/434404.htm