VLDB
Very Large Data Bases
28 August 2012 - 30 August 2012
ISTANBUL
TURKEY

Reviews For Paper

Track	Research -> March 2012
Paper ID	491
Title	Logic and Lattices for Distributed Programming

Masked Reviewer ID: Assigned_Reviewer_1

Review:

Question	
Overall Recommendation	Weak Accept (no second-chance option in this track)
Summary of the paper (what is being proposed and in what context; brief justification of your overall recommendation). One paragraph	Bloom is a dialect of Datalog for distributed programming. In the authors' previous work, only the set-monotonic Bloom programs (i.e., programs that compute sets of facts that only grow over time) are proven to allow eventual consistency (without any coordination). For example, eventual consistency could not be assumed for Bloom programs that involve aggregations since aggregations do not guarantee set-monotonicity. This paper presents an extension to the previous work which guarantees eventual consistency for broader types of Bloom programs. The latter programs perform operations (e.g., addition of numbers) that are monotonic in terms of a certain partial order. This extension also supports lattices which are analogous to the set-containment lattices considered in the authors' previous work. The paper presents a framework that can easily ensure eventual consistency, an important challenge in distributed programming. However, it is not sure how widely the presented work will be adopted in practice due to the inherent difficulty of understanding and using the presented work (e.g., understanding and designing correct lattices).
Three (or more) strong points about the paper (Please be precise and explicit; clearly explain the value and nature of the contribution).	S1. The paper presents a framework that can easily ensure eventual consistency, an important challenge in distributed programming. S2. The paper shows the utility of the presented work by presenting two case studies that implement replicated shopping cart services and causal delivery. S3. The paper provides detailed descriptions about built-in lattices and user-defined lattices as well as conversions between primitive collections and lattices.
Three (or more) weak points about the paper (Please indicate clearly whether the paper has any mistakes, missing related work, or results that cannot be considered a contribution;	W1. Due to the large overlap with the authors' previous work, this paper seems to make limited contributions. W2. It is not sure how widely the presented work will be adopted in practice due to the inherent difficulty of understanding and using the presented work (e.g., understanding and designing correct lattices). W3. Other than the two case studies in Section 5, the paper does not explicitly state other use cases.

write it so that he authors can understand what is seen as negative aspects	
Relevant for VLDB2012	YES
Novelty (Please give a high novelty ranking to papers on new topics, opening new fields, or proposing truly new ideas; assign medium ratings for delta papers and papers on well known topics but still with some valuable contribution; low novelty ratings are	With some new ideas
Significance	Improvement over existing work
Technical Depth and Quality of Content	Solid work
Experiments, Repeatability	Ok, but certain claims are not covered by the experiments
Presentation	Reasonable: improvements needed
If accepted, would you recommend a short or long talk?	Long
Would you "champion" for acceptance of the paper in a discussion with the peer reviewers?	No
Detailed Evaluation (Contribution, Pros/Cons, Errors)	D1. The code in Figure 1 is not explained in detail. Further details will be helpful. D2. The paper explains Figures 2 and 3 separately. The paper might be able to more clearly point out the main contributions by comparing these figures. D3. Consider addressing issues in W2 and W3.

Masked Reviewer ID: Assigned_Reviewer_2

Review:

Question	
Overall Recommendation	Weak Reject (no second-chance option in this track)
Summary of the paper (what is being proposed and in what context; brief justification of your overall recommendation). One paragraph	BLOOM is a datalog-based Ruby extension for declarative distributed programming with computation based on adding tuples to sets and correctness based on "confluence" -- the idea that these sets reach a fixed point. The current paper extends BLOOM to allow the user to define new monotone operations over a class of lattices, with the goal of handling a variety of monotone processes in distributed systems, such as often occur in reaching consensus.
Three (or more) strong points about the paper (Please be precise and explicit; clearly explain the value and nature of the contribution).	S1. The system clearly generalizes the datalog style programming of BLOOM in a nice and conceptually straightforward way. S2. A lot of care has gone into the implementation.
Three (or more) weak points about the paper (Please indicate clearly whether the paper has any mistakes, missing related work, or results that cannot be considered a contribution; write it so that he authors can understand what is seen as negative aspects	W1 Sections 2.2 and 3.6 on the correctness is overly informal, not giving the reader insight into what new complexity, if any, was encountered or solved by the introduction of the new lattice capabilities. In particular, "We have implemented a conservative static analysis...follows directly from the CALM theorem" - this analysis should be an algorithm, even if it is a short one. W2 The performance test is very limited, which is disappointing since the implementation of the working language seems to be a key contribution. W3 If there are no tricky parts in the static analysis, then some thought on performance optimization is needed. For example, how does "morph" interact with datalog style optimziations?
Relevant for VLDB2012	YES
Novelty (Please give a high novelty ranking to papers on new topics, opening new fields, or proposing truly new ideas; assign medium ratings for delta papers and papers on well known topics but still with some valuable	With some new ideas

contribution; low novelty ratings are	
Significance	Improvement over existing work
Technical Depth and Quality of Content	Solid work
Experiments, Repeatability	Ok, but certain claims are not covered by the experiments
Presentation	Reasonable: improvements needed
If accepted, would you recommend a short or long talk?	Short
Would you "champion" for acceptance of the paper in a discussion with the peer reviewers?	Well, not really
Detailed Evaluation (Contribution, Pros/Cons, Errors)	In 3.4, the "Because Ruby is Dynamically Typed" paragraph is concerning. If a Ruby program catches exceptions such as the type exception, does the CALM analysis still hold? What is the performance of the causal delivery and vector clock implementations?

Masked Reviewer ID: Assigned_Reviewer_3

Review:

Question	
Overall Recommendation	Reject
Summary of the paper (what is being proposed and in what context; brief justification of your overall recommendation). One paragraph	This paper introduces Bloom^L, an extension for Bloom to support arbitrary lattice structures. Bloom is a Datalog-based language used to write distributed programs which are easily verifiable to be monotonic through the CALM analysis proposed by the authors in a previous work. The original Bloom was limited to set lattices and could only detect monotonicity through the growth of such sets, which reduces the applicability of the monotonicity analysis. The contribution of this paper revolves around extending the language to support other lattice structures, such as min, max, but also to allow developers to define their own user defined lattices and methods.
Three (or more) strong points about the paper (Please be precise and explicit; clearly explain	1- The background is well introduced and interesting. Eventual consistency has been leveraged in large-scale systems such as Dynamo or Cassandra so there is a need to formalize this property and provide formal verification. 2- The problem solved by this paper is well defined, the contributions

the value and nature of the contribution).	are clear. 3- The use cases provide some concrete examples where the work is applicable.
Three (or more) weak points about the paper (Please indicate clearly whether the paper has any mistakes, missing related work, or results that cannot be considered a contribution; write it so that he authors can understand what is seen as negative aspects	1- Lack of (novel) content and insufficient contribution 2- Simplicity of CALM vs. amount of work required by user/developer 3- Bloom-L's performance hit over Bloom at little gain
Relevant for VLDB2012	YES
Novelty (Please give a high novelty ranking to papers on new topics, opening new fields, or proposing truly new ideas; assign medium ratings for delta papers and papers on well known topics but still with some valuable contribution; low novelty ratings are	With some new ideas
Rationale for novelty rating	The contribution of this paper revolves around extending the Bloom language to support other lattice structures, such as min, max, but also to allow developers to define their own user defined lattices and methods.
Significance	Improvement over existing work
Technical Depth and Quality of Content	Syntacticaly complete but with little contribution
Experiments, Repeatability	Ok, but certain claims are not covered by the experiments
Presentation	Excellent: careful, logical, elegant
If accepted, would you recommend a	Short

short or long talk?	
Would you "champion" for acceptance of the paper in a discussion with the peer reviewers?	Well, not really
Detailed Evaluation (Contribution, Pros/Cons, Errors)	1- There is not enough content in this paper. Many sections were very repetitive, especially section 2. Some sections were most concerned with implementation details or statistics about lines of code. For instance, section 3.5.1 is simply syntactic sugar and not really necessary to the understanding of the work. 2- The contribution of this paper specifically is insufficient. While the overall goal of eventual consistency is interesting, this goal has already been achieved in a previous paper by the authors. The extension shown here is just about support other types of lattice and does not really add new insights to the monotonicity property. 3- The CALM analysis seems simple, given the amount of work the developer has to do to correctly define the lattice structures and manually declare monotone, morphisms, non-monotone functions. It seems that the basic principle of the analysis is recursive: a component monotonic if its sub-components are monotonic. If that is the case, why not run a static analysis at compile-time? A control flow graph should be able to determine whether any lattice structures are growing in a monotonic fashion. The analysis should be the least disruptive to the developer as possible, and not impose demands such as programming in a different language. Figure 5 shows that above the ~95 percentile, both QW(4) and MDCC outperforms QW(3). This result is counter-intuitive and never explained. Also the performance evaluation is quite subjective. MDCC is essential double the latency of QW(3): this does not seem insignificant. 4- The evaluation shows that Bloom^L has additional overhead over Bloom. This is troublesome because Bloom^L is not functionally more powerful than Bloom. The extension is only needed to support the analysis. I doubt developers are willing to sacrifice run-time performance for the sake of formally verifying the monotonicity of their program. This suggests that a compiler is required to convert Bloom^L programs into Bloom.