سمینار

تمرین سوم

مینا فریدی
810100430

سوال: بـرای هـر یک از بخش های جمع بـندی و نتیجه گیری زیر که از مقالات مختلفی انتخاب شـده انـد، ابتدا اجزاء را مشخص نمایید. سپس چـنانـچه تـصور می کنید که جـزئی در آن مشاهده نمی شود، با ذکر دلایل خود، جزء یا اجزای غایب را نام ببرید.

پاسخ: بخش‌های جمع‌بندی و نتیجه گیری موارد زیر می‌باشند:

- **هدف یا فرضیه اصلی**
- **یافته‌ها**
- **علل نتایج**
- **محدودیت‌های مطالعه**
- **نتایج جانبی**
- **تحقیقات آتی**

در ادامه این بخش‌ها را به صورت رنگی مشخص کرده و در صورت عدم وجود با بیان دلیل توضیح می‌دهیم.

مقاله اول

We have presented a distributed approach to stream reasoning that uses a novel intervalbased representation of LARS semantics, aiming to improve storage and communication performance between reasoning components. (هدف یا فرضیه اصلی) A given stream-stratifiable plain LARS program is automatically decomposed into layers whose evaluation depends only on the results of lower layers. The layers are mapped to a generated network of stream reasoners that can process the program in a distributed way. We have described an implementation of this architecture in a system that can accommodate various stream reasoners for the components. (علل نتایج) The evaluation results indicate that the system outperforms existing stream reasoners for the same LARS fragment on a realistic monitoring problem. Finally, the scalability of the novel system was shown by a enchmark on a chained n-Queens completion problem, which could not be solved easily by previous LARS reasoners. (نتایج)

Outlook. The work that we have presented can be continued in several directions. On the one hand, the repertoire of window operators can be enriched and made available in the implementation; in particular, aggregates that can be viewed as particular window functions may be provided, but furthermore also added as first-class citizens to the LARS

framework, for both traditional LARS streams and interval streams. As regards the distributed computation approach, in particular the following issues are on our agenda: (i) to investigate issues related to distributed computation, such as time synchronization or liveness, in order to ensure the correctness of answer streams that are output by the system; and (ii) to improve incremental reasoning techniques with on-the-fly grounding; and (iii) to provide distributed answer set computation that supports full plain LARS, i.e., programs that are not stream-stratified. (کارهای آتی)

این مقاله به ارائه یک متد پرداخته است و محدودیتی را در تحقیق را بیان نکرده است چرا که از ابتدا متد را ابداع نموده. همچین چون به نتیجه گیری جانبی‌ای هم دست پیدا نکرده این این بخش نیز در قسمت جمع‌بندی مقاله موجود نمی‌باشد.

مقاله دوم

- **هدف یا فرضیه اصلی**
- **یافته‌ها**
- **علل نتایج**
- **محدودیت‌های مطالعه**
- **نتایج جانبی**
- **تحقیقات آتی**

The paper proposed a demonstration that shows that the video stream processing pipelines can be federated via autonomous stream fusion agents (هدف یا فرضیه اصلی). These agents are facilitated by stream fusion kernels that can adaptively utilize its resources to alleviate the process bottleneck at a central server which(یافته‌ها) is caused by bandwidth and computing overloads from distributed video cameras.(علل نتایج) This opens up several interesting research directions. The first direction is to study the overhead and trade-off of the proposed federated execution mechanism. The second direction is how to build a general distributed event regonition model for this setting. The third direction is how to build the cost model and optimisation algorithms for such a system. The final direction is how to integrate such video streams with other kind sensor sources from autonomous vehicles such as Lidar and rada.(تحقیقات آتی)

این مقاله نیز محدودیت‌های موجود در تحقیق را بیان نکرده است زیرا حرفی از محدودیت‌ها نزده است. نتایج این تحقیق کمتر پرداخته شده است و به نظر می‌آید که نتایج جانبی هم اصلا وجود ندارد. در کل بیشتر به کارهای آتی پرداخته است.

مقاله سوم

- هدف یا فرضیه اصلی
- یافته‌ها
- علل نتایج
- محدودیت‌های مطالعه
- نتایج جانبی
- تحقیقات آتی

We presented I-DLV-sr, an ASP-based stream reasoner relying on a tight interaction between I 2 -DLV and a Flink application, that in the experiments showed good performance and scalability. I-DLV-sr is easily extendable by design; hence, we plan to add the support to additional language constructs while extending tests over new real-world domains. Furthermore, we plan to study proper means to extend the language for the management of noise and incompleteness, and further move towards a more complete SR reasoner (Dell'Aglio et al. 2017).

این مقاله کارایی نتایج را بیان کرده اما دلایل آن را توضیح نداده است و درباره محدودیت‌های مطالعه نیز سخنی نگفته است. و نتایجی را به صورت ضمنی نیاورده است.

مقاله چهارم

- هدف یا فرضیه اصلی
- یافته‌ها
- علل نتایج
- محدودیت‌های مطالعه
- نتایج جانبی
- تحقیقات آتی

We are observing an impressive increase of the speed of data production and consumption.

In this paper, we explained how stream reasoning aims at providing methods and tools to perform sophisticated analyses of such data. In the beginning, stream reasoning grew with the idea of building such analyses on top of logical and deductive inference. DSMS, CEP and SemWeb offered solid starting points to kick off the research Through the years, we have observed the creation of languages, techniques and frameworks. Those studies pushed stream reasoning in a broader area, introducing reasoning techniques beyond the deductive ones. Semantic Scholar and Google Scholar count more than a 1000 articles containing "stream reasoning",5 published in different areas, from semantic web to artificial intelligence. However, there is still a lot of research to be done. In Section 3, we presented the main directions over which stream reasoning research can continue. Stream reasoners should offer richer query languages, which include a wider set of operators to encode user needs, and the engine to evaluate them. Reasoning took a more generic connotation, and now it includes inductive reasoning techniques in addition to deductive ones. This trend will grow, combining different techniques to overcome their respective limits. Solutions need to be engineered in scalable frameworks, i.e., they must be able to integrate and reason over huge amounts of heterogeneous data while guaranteeing time requirements. And it will be important to fill the gaps between theoretical models and reality, making stream reasoning solutions robust and able to cope with issues such as noise and heterogeneity. In parallel, it will be important to identify real problems and scenarios where stream reasoning may be a solution. Internet of Things and Industry 4.0 are examples of areas where to apply stream reasoning results. Moreover, it is necessary to develop benchmarking and evaluation activities, to compare and contrast the current solutions. Results obtained up to now are important. In addition to the publications, some of the mature solutions were exploited in real scenarios, such as social media analytics and turbine monitoring. We should get inspired by such results, and see them as the foundations to build new research and to reach new ambitious achievements, to reach the goal of: making sense in real time of multiple, heterogeneous, gigantic and inevitably noisy and incomplete data streams in order to support the decision process of extremely large numbers of concurrent users

این یک مقاله‌ی survey است و در واقع می‌توان اکثر توضیحات آن را نتایج یا نتایج جانبی در نظر گرفت. محدودیت های مطالعه می‌تواند مطالعه کردن منابع بسیار زیاد (1000 مقاله stream reasoning) نیز باشد.

مقاله پنجم

- **هدف یا فرضیه اصلی**
- **یافته‌ها**

- علل نتایج
- محدودیت‌های مطالعه
- نتایج جانبی
- تحقیقات آتی

In this paper, we have discussed caching techniques used in modern ASP solvers and presented two approaches to the management of learned constraints based on reinforcement learning. The evaluation results that we presented indicate that proper reuse of data obtained while solving one instance from a data stream can significantly improve the performance of modern solvers while solving subsequent instances, and hence of ASP-based stream reasoning engines on top of them. Moreover, the experiments provided support for the findings of previous research on the application of truth maintenance system techniques in stream reasoners like TICKER (Beck et al. 2017). Progress saving – a JTMS-like caching strategy – appears to be very useful in monitoring applications of technical systems. Since massive changes or failures are rarely observed in such environments under normal conditions, data delivered by such systems in subsequent time points is highly interrelated. Therefore, a solver using progress saving can easily restore consistent parts of a previous model and focus only on repairing of a rather small number of remaining unsatisfiable assignments. However, current progress saving methods are less versatile in comparison to JTMS techniques when the number of possible constants in a program is large. In such situations, they usually cannot select which literals must be removed from the cache as it grows in size. This finding opens an interesting direction of future research, especially in conjunction with predictive overgrounding techniques (Calimeri et al. 2019).

The positive effect of caching of learned constraints was observed in situations when data in the input stream caused many inconsistent assignments in the existing model. In such situations learned constraints were able to provide valuable information to the solver that could be fruitfully used to reduce its search time. However, our findings also showed that the application of reinforcement learning in this area must be studied in more detail. In our future work, we are going to focus on automated identification of reward functions that work best for a particular encoding in the LARS language and we shall consider experiments with highly dynamic domains, such as cooperative intelligent transport systems.