# Assembly Line Analytics

Christan Grant, Daisy Zhe Wang, Joseph N. Wilson
Department of Computer & Information Science & Engineering
E403 CSE Building
Gainesville, Florida, 32612
{cgrant,daisyw,jnw}@cise.ufl.edu

## 1. INTRODUCTION

The amount of data generated by sensors, machines and people is increasing and shows no signs of slowing down. To extract understanding and meaning from this data researchers have developed several algorithms and architectures. To deal with the deluge of data researchers create algorithms specially designed to work with large-scale architectures. These new algorithms, when employed in productions are expected to perform faithfully as the algorithms scale. Analysts and engineers often employ these large-scale algorithms in production over large scale tasks. Once these algorithms are running in productions, correcting or updating parameters is impossible or poorly supported. Analysts must re-push their code to integrate any changes or updates breaking the natural progression of the algorithm.

When addressing errors in analytic pipeline analysts are often too late or the self-updating algorithm lags behind the current algorithm. Changes in order to address changes to the algorithm updates and humans could preemptively make changes and updates in <expencansy> of new signals.

Given these issues with the current world of large-scale analytic production stacks. In this paper we propose a vision for assembly line analytics systems.

In the following sections we ...

## 2. PROBLEM STATEMENT

Given the increasingly massive amounts of streaming data and the scaling algorithms employed to monitor the progression of the algorithms; simply deploying methods to extraction inferences and conclusions from the stream is not enough. As the data grows, discovering micro anomalies of interest is increasingly difficult. ■ Show graph that as data grows, the time to extract detailed information from data increases> As the data grows the integrity of analytic systems may decrease to because of the increasingly variety of data.

We introduce a philosophy of Assembly line analytics. Assembly line analytics is the evolution of query-driven algorithms from large-scale streaming systems. In order to keep

A user cannot be blocking in the loop because it will cause delays in the stream. Instead, an analysts should be 'over the loop' much like an assembly line worker so there is no bottle neck in the algorithms.

Any user interactions with online algorithms must affect the systems in the way a user expects. This means a delay in user intervention or any latency in user actions should be resolved by the system.

Multiple users should by able to analyze and affect the pipeline independently of each other.

Two observations: 1) Analysts only care about a subset of data 2) Analysts have an innate understanding of the success of algorithms that is computationally expensive for algorithms to duplicate.

## 3. RELATED WORK

(Move to right before conclusion?)

Approximate early query answer (DBO, BlinkDB) - This work is complimentary. However users are not able to interrupt or change queries once they are executed.

Interactive analytics (MSR Work) . . . - These techniques are complimantary but they often block awaiting user respones.

Query-Driven Analytics (Me) . . . - This work is complemtarty and can be used at a micro-level but in its current form does not allow users to move between queries to explore the space of all queries.

Video Games . . . - Many video games include an AI that plays along with the user. The user can even switch in and out of activing playing situations. This work can use the idea of humans and AI cooperating with humans assisting only when necessary. <Add the example of sports games> This work would allow the Human to be a passive expert only intervening when necessary.

## 4. ASSEMBLY LINE ANALYTICS

1) Let users exam and check the progress of the algorithms at any point in the life of the algorithm. 2) Users should be able to intervene on the in the running of the algorithm to improve/redirect goals 3) Multiple users may be added to concurrently monitor the progress of algorithms.

## 5. ARCHITECTURE

*DanaC'13,* June 23, 2013, New York, NY, USA
Copyright 2013 ACM 978-1-4503-2202-7/13/6 ...$15.00.

In this work we describe a first version to follow the principles of assembly line analytics.

■Figure with two parts 1) Interactive interface and dashboard, 2) Intelligent layer to translate user actions to a middle language of action the algorithm understands. 3) The back-end, running a streaming data algorithm that will perform actions in response of new data or user action. ■

We developed a prototype using the Scala and the play framework.

We use Akka Actors to develop the algorithms for scalability, redundancy and atomics. Actors have guaranteed message delivery. Actors Systems can be deployed to have redundancy and durability. Streaming actors systems are fast and are able to scale to a massive number of nodes.

We should not user larger scaler systems because we need fine control over micro portions of the data.

## 6. EVALUATION

Micro: - human interference latency (latency with helping progress or helping to undo/redo) - human reation time to difference stimuli - What is a threshold between believing human intervention and being skeptical of human effort - What do humans think about the interface?

Macro: - Accuracy with humans intervention w/o human without - Accuracy with a gradient of human interventions (a lot of interventions to a little) (Do humans slow down progress?) - Do multiple analysts help? Is there an optimal number of concurrent analysts? - Does suggesting problem areas help?

## 7. CONCLUSION

We discussed . . .