Inhaltsverzeichnis

1. Einleitung

In der Produktion fallen neben viele sogenannte Qualitätsdaten an, diese sind logisch stark mit Produktionsdaten in ERP-Systemen verknüpft. Sie sind logistisch, kostenrechnerisch oder buchhalterisch relevant. Diese Sensordaten produzieren hohe Datenmenge, im besonderen durch eine Vielzahl an Sensoren. Dadurch gibt es eine sehr heterogenen Datenbestand. Da der Bestand an Sensoren mit der Unternehmensgröße wächst und es ständig zu Änderungen durch das Hinzukommen und Enfallen von Messinstrumenten kommt braucht man eine Datenstruktur die sich diesem Wandel anpassen kann.

Ziel dieses Projektes war es das Problem der Datenspeicherung mit 4 Lösungsansätzen zu lösen. Dabei sollte eine proprietäre Datenstruktur, eine universell relational Struktur, eine Schlüssel-Werte-Datenbank und eine dokumentenorientierte Datenbank zum Einsatz kommen. In allen Lösungen sollten Daten konsistent gespeichert werden. Weiterhin sollten verschiedene Analysen möglich sein und auch auch das Darstellen der Analyseergebnisse.

2. Problemstellung

In dem konkreten zu implementierenden Beispiel ging es um fünf Maschinen beteiligt an einem carbonteilfertigungs Prozess. Dabei werden teure Carbonfasern mit einem günstigen Trägerstoff vernäht. Ziel dabei ist es eine bessere Handhabarkeit bei geringeren Herstellungskosten bereitzustellen. Die Maschine erzeugt einen Eingangsdatensatz mit 23 Werten und einen Ausgangsdatensatz mit 16 Werten. Dabei werden neben der Seriennummer, die spezifisch für ein Werkstück (Teil) ist, der Fertigungsauftrag und die Ladungsträgernummer gespeichert. Zu einem Fertigungsauftrag gehören mehrere Teil und ein Ladungsträger kann ebenfalls für mehrere Werkstücke verwendet werden.

Der Einlesealgorhytmus stellt etwas vereinfacht dar wie die Datensatzerstellung mit der Fertigung auf der Maschine zusmmenhängt. Sobald das Teil korrekt in die Maschine eingelegt wurde wird der Eingangsdatensatz erstellt. Nach der eigentlichen Fertigungsarbeit auf der Maschine wird das Teil überprüft. Dabei entsteht der Ausgangsdatensatz. Falls die Prüfung nicht erfolgreich war wird das Produkt erneut überprüft. Danach verlässt das Teil die Maschine, falls nun Probleme mit dem Werkstück gefunden wurden wird das Werkstück erneut in die Maschine gegeben, falls nicht

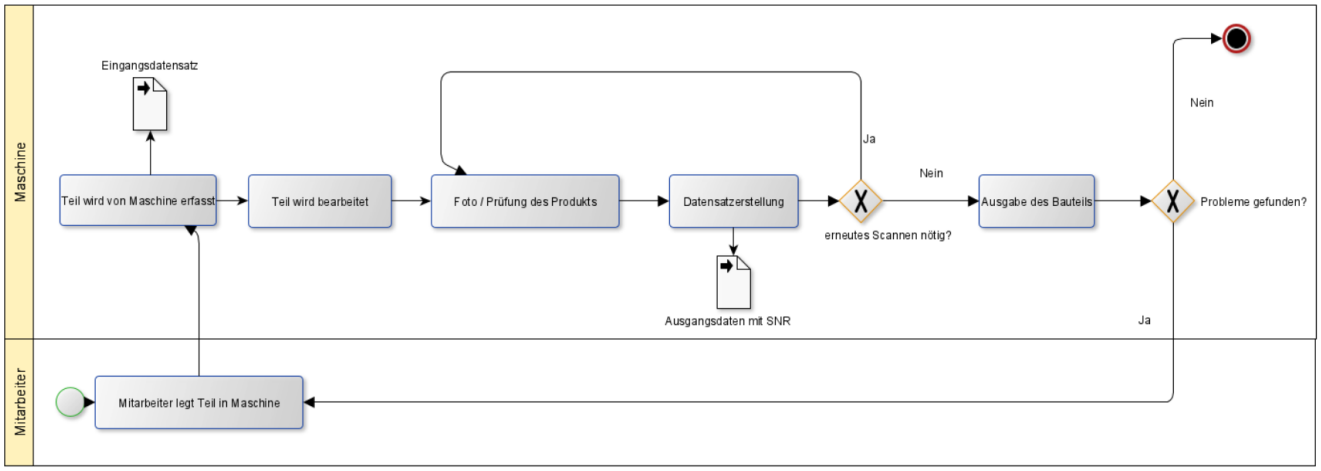


Abbildung 1. Einlesealgorhytmus

3. Anforderungsanalyse

4. State of the Art Lösungen

4.1. Hinführung

Die Recherche zum Thema Speicherung von Daten im IOT Bereich erwies sich schwieriger als gedacht.

4.2. Amazon Webservice

4.3. Iconics Hyper Historian

Zeitreihen -→ gut für wenige Sensordaten ständige Wertmessung Speicherung in einer proprietäre Datenbank und das Änderungen auf Dateieben darauf schließen Sortierte Hashmap oder ähnliches -→ Anpassung der Schlüssel und neu Sortierung des Baumes

4.4. Fab Eagle Cloud

Die temporäre Ablage von Daten in performanten Puffern mit Hilfe von Redis ist möglich.

5. Vorstellung der Lösungen

5.1. Proprietäre Datenstruktur

5.1.1. Einführung in Datenbankstruktur

Datenbankvorstellung

5.1.2. Entwurf

Erstes Datenmodell

5.1.3. Implementierung

Implementierung der Struktur

Implementierung der Datenloader

Implementierung der Analysen

```
@app.route('/')
def home():
    return render_template("inline.html")
```

Evolution des Datenmodells

5.2. Universelle relationale Struktur

5.2.1. Einführung in Datenbankstruktur

Datenbankvorstellung

5.2.2. Entwurf

Erstes Datenmodell

5.2.3. Implementierung

Implementierung der Struktur

Implementierung der Datenloader

Implementierung der Analysen

```
@app.route('/')
def home():
    return render_template("inline.html")
```

Evolution des Datenmodells

5.3. Dokumentenorientierte Datenbank

5.3.1. Einführung in Datenbankstruktur

Datenbankvorstellung

5.3.2. Entwurf

Erstes Datenmodell

5.3.3. Implementierung

Implementierung der Struktur

Implementierung der Datenloader

Implementierung der Analysen

```
@app.route('/')
def home():
    return render_template("inline.html")
```

Evolution des Datenmodells

5.4. Schlüssel-Werte-Datenbank

5.4.1. Einführung in Datenbankstruktur

Datenbankvorstellung

5.4.2. Entwurf

Erstes Datenmodell

5.4.3. Implementierung

Implementierung der Struktur

Implementierung der Datenloader

Implementierung der Analysen

```
@app.route('/')
def home():
    return render_template("inline.html")
```

Evolution des Datenmodells

6. Analysen

6.1. Analyse 1

Die erste Analyse beschäftigt sich mit der Taktung pro Artikel. Dabei wurde die Differenz zwischen Eingangs- und Ausgangsdatensatz gemessen. Danach wurde pro Fertigungsauftrag gruppiert und das Minimum, das Maximum und der Durchschnitt ermittelt. Die Ergebnisse wurden auf eine Stunde beschnitten um nicht zu sehr von Ausreißern beeinflusst zu werden. Die Alternative wäre gewesen nur die Zeiten zu nutzen, die in der Fertigungszeit des Unternehmens liegen, da aber die Auswertungen nur als Grundlage zum Vergleich der Datenbanklösungen dienen sollte wurde das nicht implementiert.

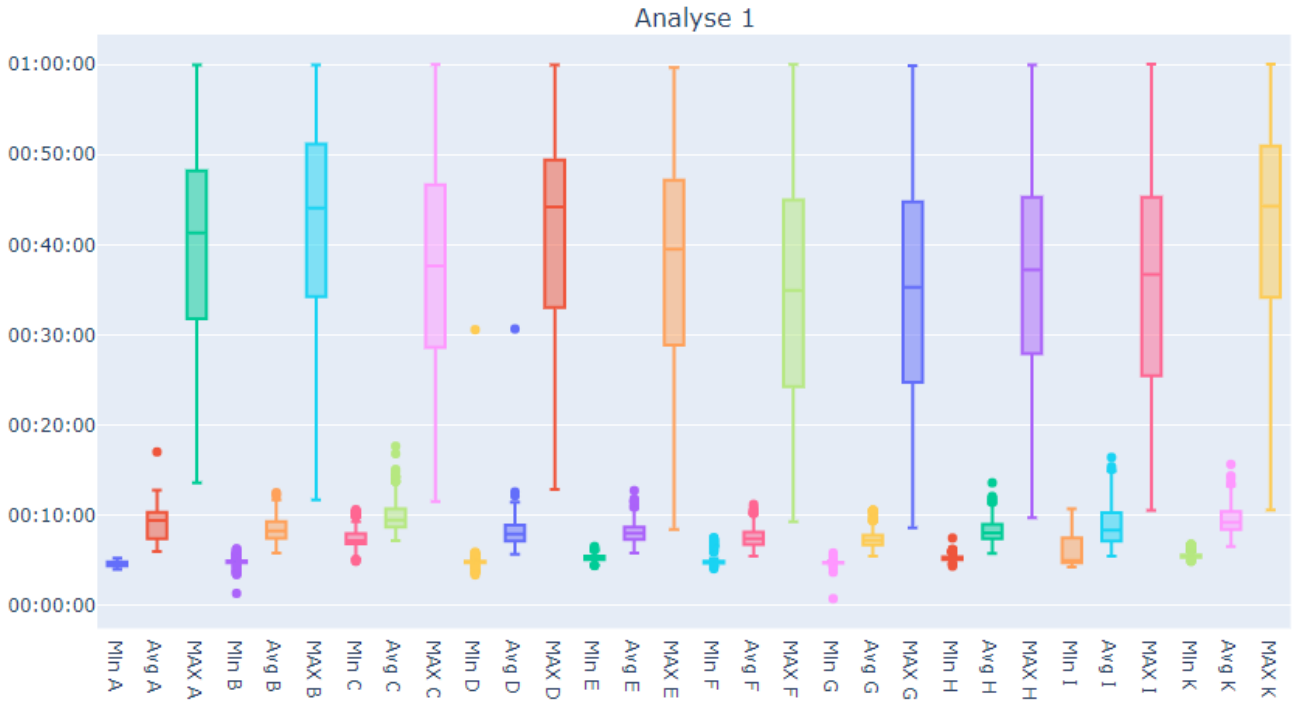


Abbildung 2. Analyse 1

6.2. Analyse 2

Diese Analyse beschäftigt sich nur mit Ausschuss Artikeln, also Werkstücke die mehrmals in die Maschine eingelegt worden sind, also deren SNR n In-Datensätze besitzt. Dabei wurde zunächst die Differenz zwischen Ende des fehlerhaften Vorgangs und Beginn des neuen Vorgangs gemessen. Auch hier wurde wieder Minimun, Maximun und der Durchschnitt berechnet. Aber über die Gruppierung Teil. Auch wurde der Anteil an Fehlerhaften Bearbeitungen wurde ermittelt.

Abbildung 3. Minimun

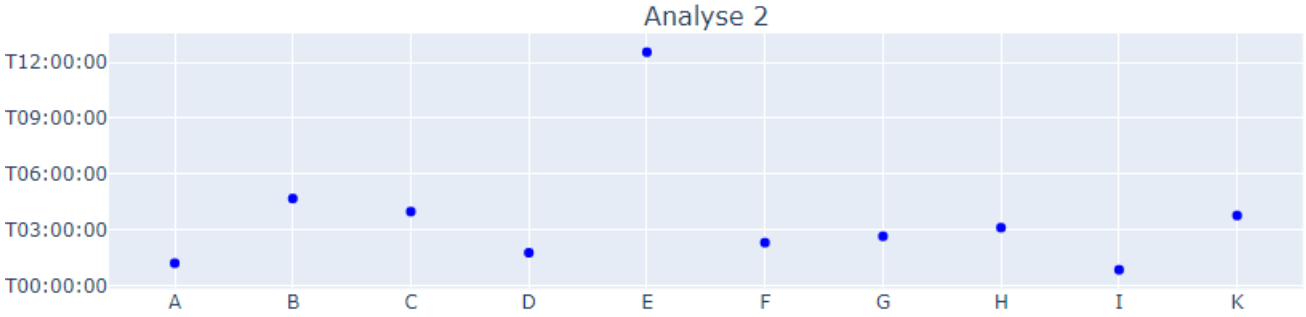


Abbildung 4. Durschschnitt

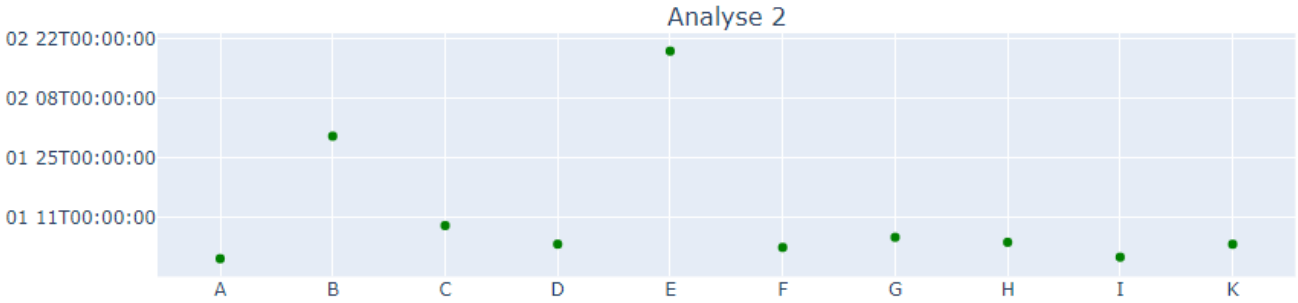


Abbildung 5. Maximum

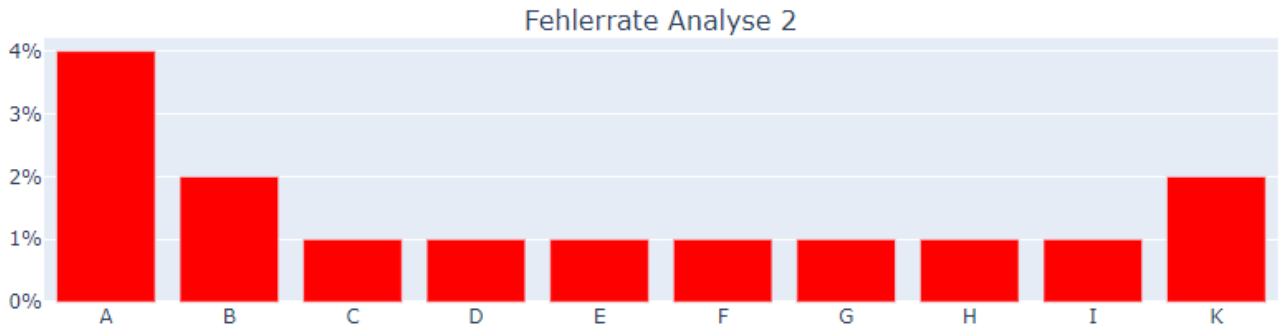


Abbildung 6. Fehlerrate

6.3. Analyse 3

Die dritte Analyse nutzt die gleichen Daten wie Analyse 1. Nur mit veränderter Darstellungsform. Ziel war es hierbei Ausreißer zu finde, deshalb wird nun nicht mit Boxplots gearbeitet, sondern mit einem Streudiagramm. Damit ist es besser ersichtlich, wo entfernte Ausreißer zu finden sind.

Abbildung 7. Minimun

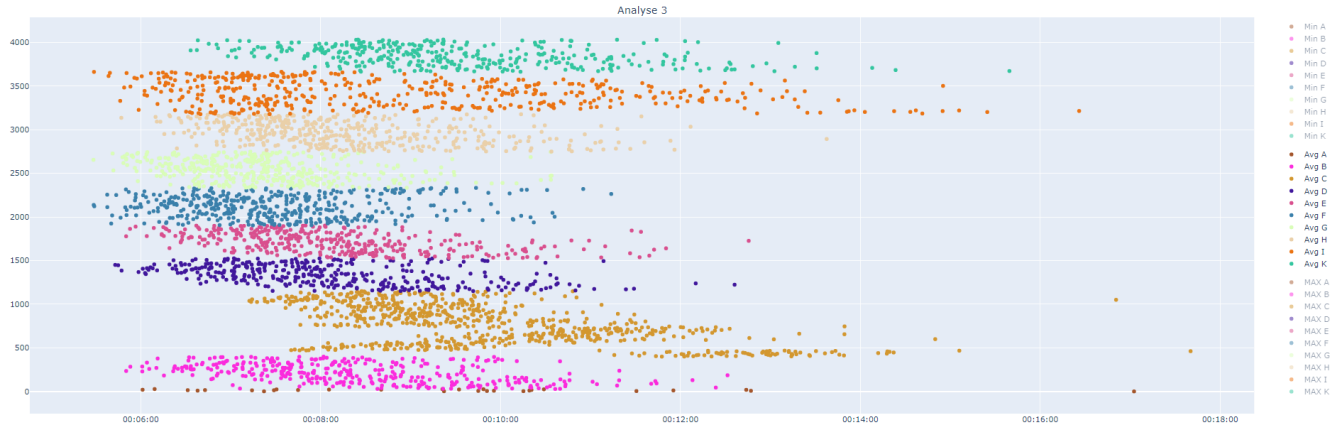


Abbildung 8. Durschschnitt

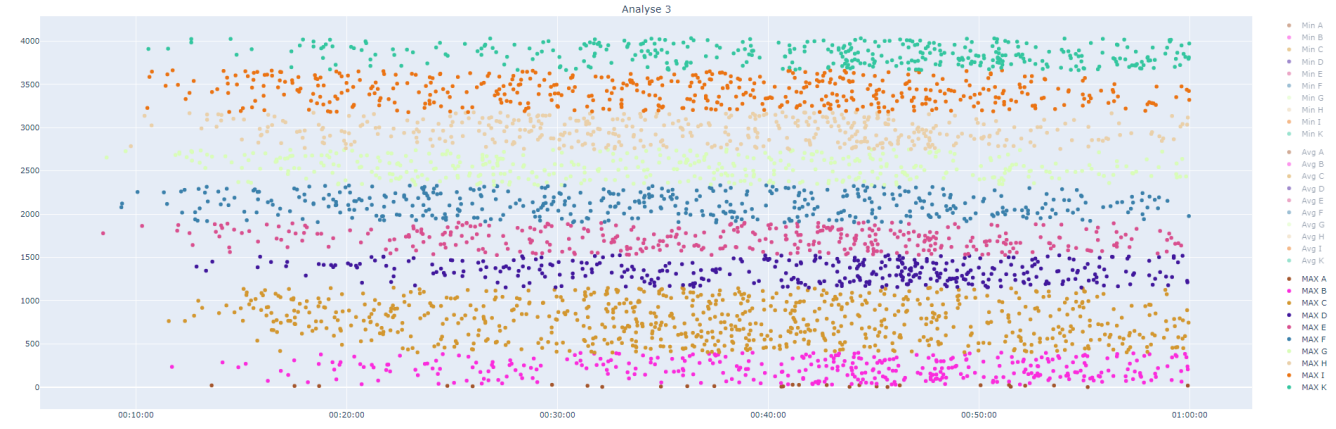


Abbildung 9. Maximum

6.4. Analyse 4

Die vierte Analyse zeigt die Nutzungszeit der Ladungsträger. Dabei ist die Nummer an der Y-Achse irrelevant und stellt jeglich die Datensatznummer dar, denn in den Ladungsträgernummern gab es eine zu große Streuung um diese adequat darzustellen.

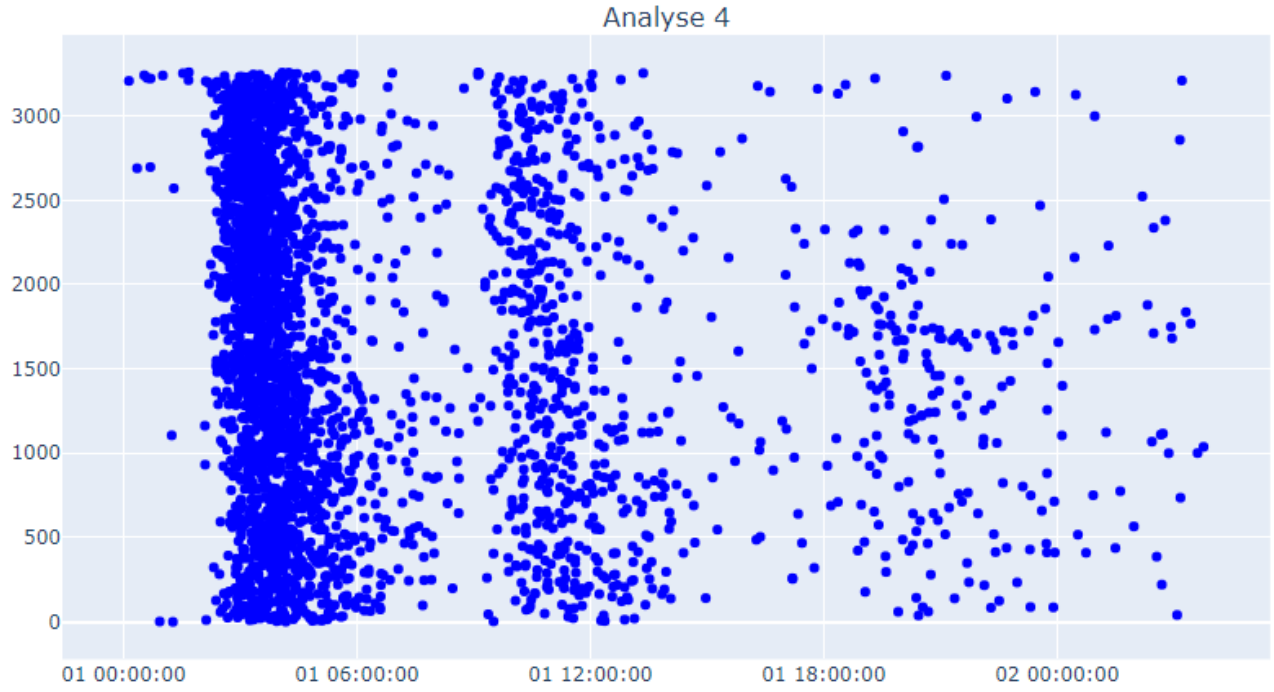


Abbildung 10. Analyse 4

6.5. Analyse 5

Die fünfte Analyse ist analog zur ersten Analyse, nur das dieses mal die Zwischenaggretion nicht pro Fertigungsauftrag sondern pro Ladungsträger erfolgte. Danach wurde die Daten wieder nach Teil sortiert und jeweils ein Boxplot für Minimum, Maximum und den Durchschnitt gezeichnet.

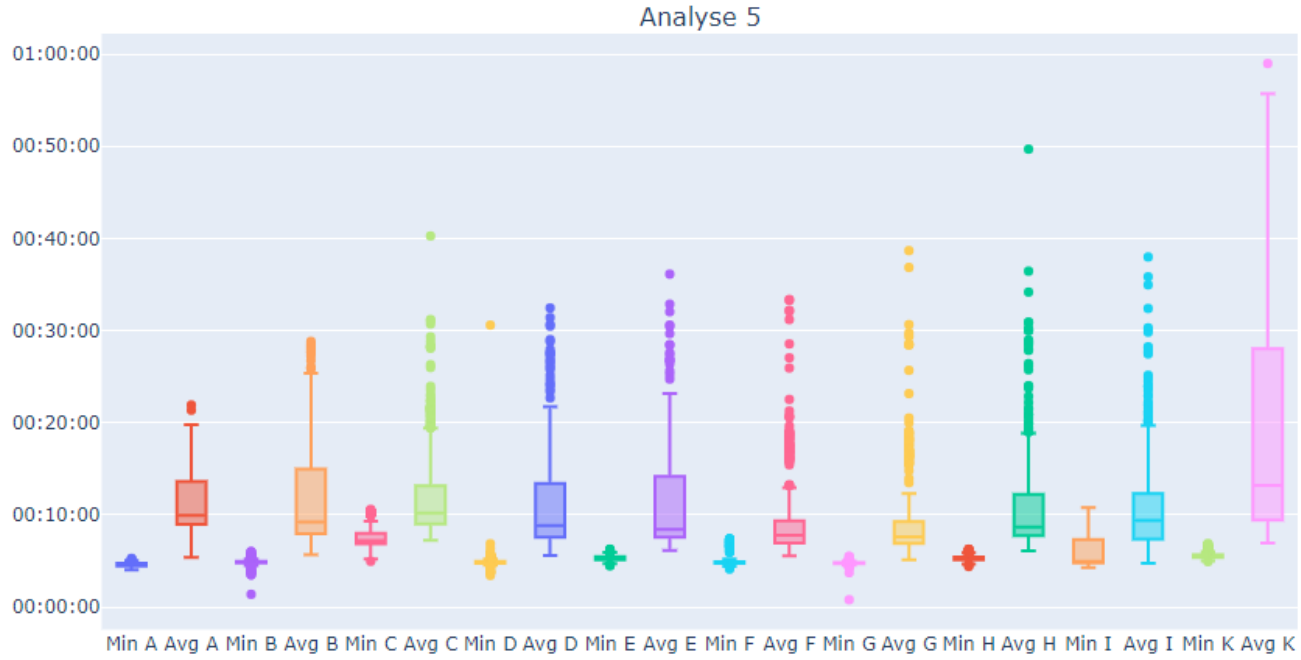


Abbildung 11. Minimun und Durschnitt

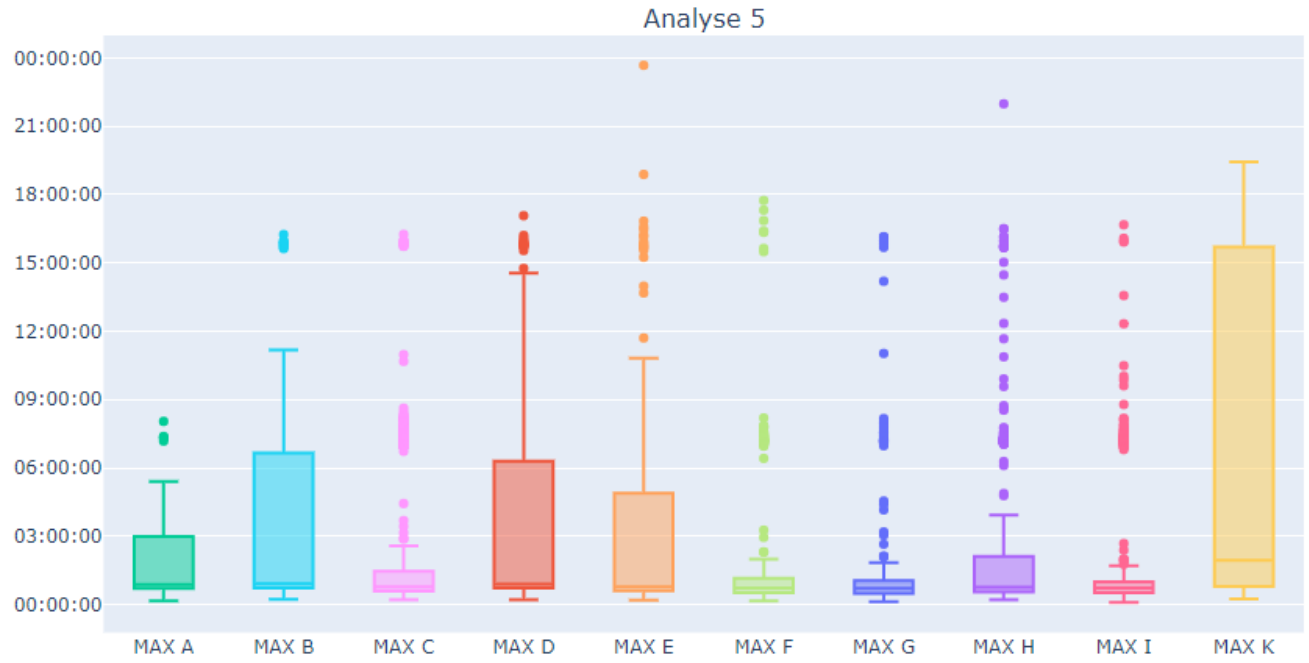


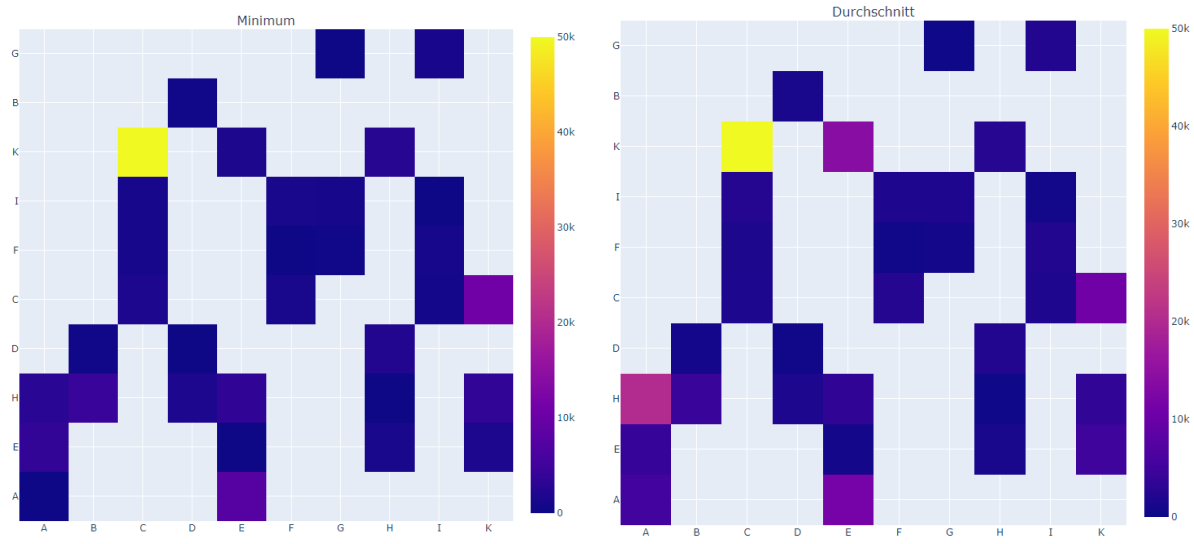
Abbildung 12. Maximum

6.6. Analyse 6

Die zunächst vom Themensteller geforderte sechste Analyse hatte sich als obsolet herausgestellt. Denn sie sollte die Umrüstzeiten von Teil und Ladungsträger analysieren. Der Informationsgehalt wäre aber gering gewesen, da ein Ladungsträger meist nur einmal

6.7. Analyse 7

Die letzte Analyse zeigt die Umrüstzeiten zwischen den Teilen. An der Y-Achse stehen die Teile von denen auf die Teile die an der X-Achse umgerüstet wurde. Je heller die Farbe ist, desto höher ist der Wert. Hier wurde Minimun, Maximum und der Durchschnitt pro Umrüstvorgang bestimmt.



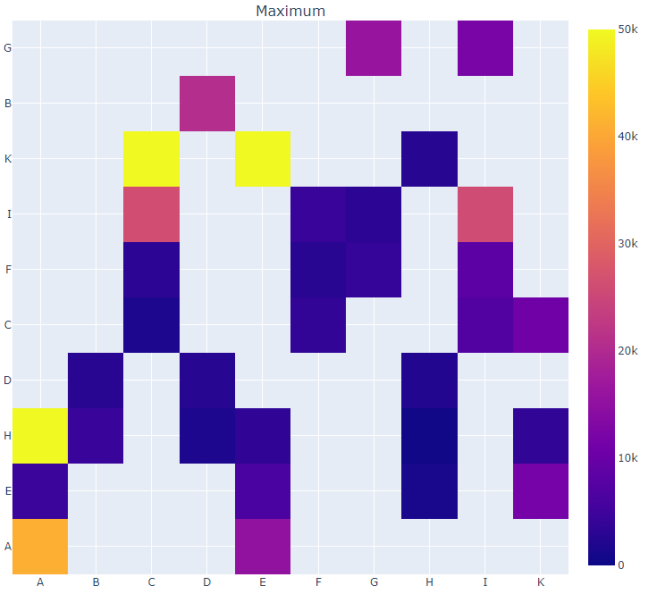
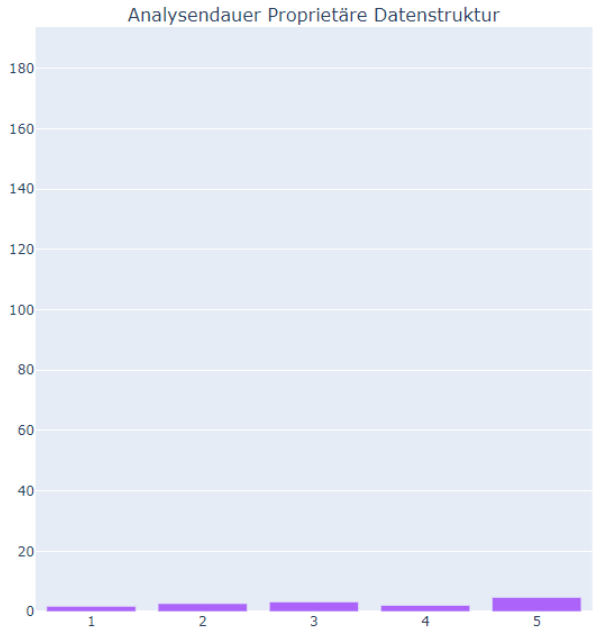


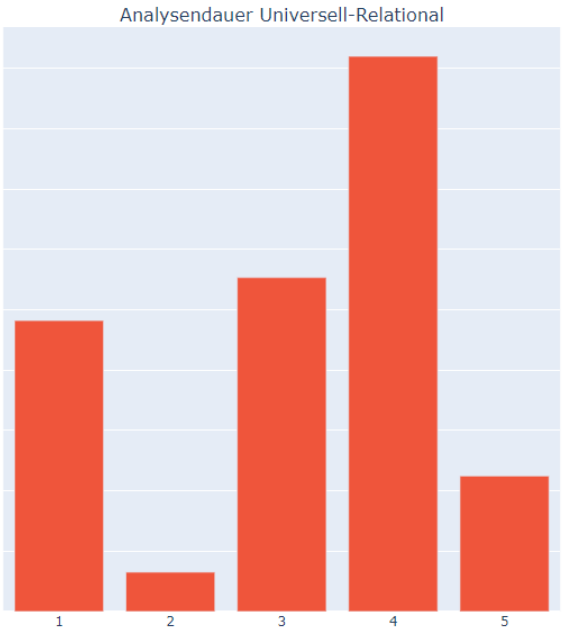
Abbildung 13. Maximum

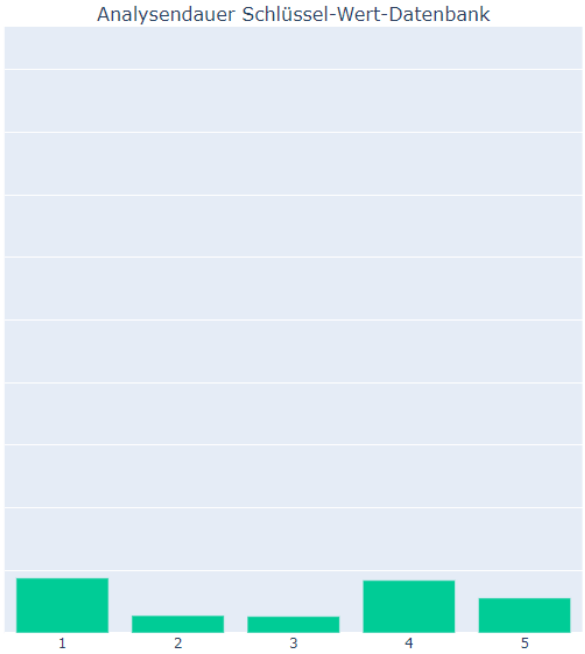
7. Lösungsvergleich

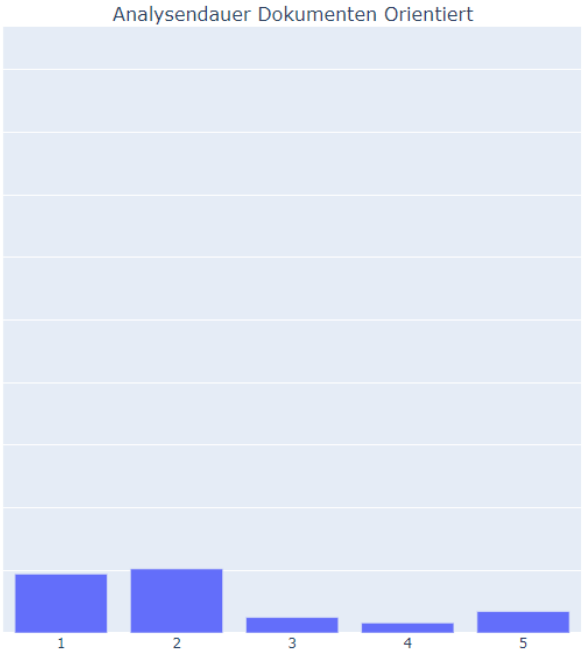
7.1. Übersicht

7.2. Performance









8. Zusammenfassung und Ausblick

9. Quellen