Improving sensitive data generation and limiting event logs variety loss

The added features aim to decrease the number of events removed from the collection (in order to follow the t-closeness and k-anonymity constraint), this is made possible by generating synthetic data, the addition of synthetic data made in a correct manner does not decrease the privacy of users, it enhances it indeed. Another important feature consists in improving the generation of what's considered sensitive data (the duration of the actions described in the log). The feature optimizes the generation of synthetic data, improving their accuracy to made them as indistinguishable as possible from the real ones, in this way the user privacy can be preserved, reducing the tradeoff between privacy and utility.

The extension is contained in the PRETSA folder. pretsa.py has been modified, the modified/added methods have been extensively commented.

demo.py can be run and will show the benefits of generating synthetic data to avoid pruning as required by the PRETSA paper to keep the k-anonymity and t-closeness values within the limits. To run the demo that will show graphs (it could take some times since we test and show multiple configurations):

```
cd PRETSA
pip3 install numpy==1.24.2 pandas==1.1.05 scipy
pip3 install matplotlib>=3.7.0 anytree
python3 demo.py
```

The extension includes three main components:

- Synthetic data generator
- Enhanced annotation generator
- Logging and reporting function

Synthetic data generator:

When the k-anonymity is slightly lower than the k-anonymity required in any node in the prefix tree, then the tree is pruned causing all the logs and events registered in that branch to be modified and merged into another branch with noticeable modifications, leading to a loss in data in terms of quantity, quality and variety. The synthetic data generator acts in specifics cases to avoid pruning the tree and artificially raising the equivalence set k-anonymity by injecting synthetic data (more details in the code). The DFG (Directly Follows Graphs) analysis, often used on this kind of data (as stated in the PRETSA paper), are based on analyzing the sequence of actions, their characteristics and their correlation with sensitive data. Lost in activity sequences caused by pruning could lead to worst results, it is therefore important to keep the quality and variety of data high without compromising the privacy.

Enhanced annotation generator:

The enhanced annotation generator allows us to generate annotation data closer to the one available in the original data. This is important in order not to give any hints on what may or may not be the synthetic data. As better stated in the code there is a strong correlation between the actions coming before the log we want to add the annotation to and the annotation value itself, this is testified by the fact that tha paper states that a DFG algorithm will be run on this data, and that kind of algorithm is extremely sensitive to correlations between the sensitive data and the action sequences. The enhanced annotation generation benefits are not directly showed in the graphs since it is extremely hard to quickly show it. The algorithm and the function documentation will help to understand its usefulness.

Do not use the IDE run since I've noticed some issues in rendering the matplotlib graphs. The dataset used is the one available in good environmental, other datasets struggled to run and showed incompatibilities with the code.