http://e-conf.usd.ac.id/index.php/USDB
Universitas Sanata Dharma, Yogyakarta

PENINGKATAN KUALITAS *HIGH-UTILITY ITEMSET* MENGGUNAKAN PENDEKATAN *SWARM INTELLIGENCE* PADA KASUS ANALISIS KERANJANG BELANJA

Ridowati Gunawan [1,*]
[1]*Program Studi Informatika, Universitas Sanata Dharma*
**Email : rido@usd.ac.id*

Abstrak

Terdapat perubahan paradigma dalam pencarian pola pada keranjang belanja, awalnya hanya memperhatikan *itemset* yang sering muncul secara bersamaan, sekarang tidak sekedar *itemset* yang sering muncul tetapi juga memperhatikan nilai *utility* dari *itemset*, yang dikenal sebagai *high-utility itemset.* Kuantitas dan bobot setiap item menjadi faktor utama untuk mendapatkan *high-utillity itemset*. Akan tetapi ruang pencarian *high-utility itemset* sangatlah luas oleh karenanya diperlukan berbagai teknik untuk mendapatkan *high-utility itemset* agar hasil yang diperoleh mendapatkan hasil yang optimum. Penggabungan metode pencarian *high utility itemset* dan *swarm intelligence* digunakan untuk meningkatkan kualitas dari *high-utility itemset.* Pendekatan *swarm intelligence* yang digunakan adalah *particle swarm optimization* yang memiliki empat parameter utama yaitu populasi awal, *inertia weight, coefficient acceleration* dan *velocity clamping.* Pengukuran kualitas *high-utility itemset* yang ditinjau dari jumlah itemset, total itemset, *running time* dan penggunaan memori. Hasil percobaan yang dilakukan terhadap *dataset* keranjang belanja diperoleh bahwa *total utility* mengalami peningkatan dan faktor yang paling mempengaruhi kualitas pencarian *high-utility itemset* menggunakan pendekatan *swarm intelligence* adalah *inertia weight.* Selain itu penggunaan pencarian tanpa memasukan nilai *minimum utility* meningkatkan kualitas *high utility itemset* ditinjau dari total *utility* yang diperoleh, yaitu mengalami peningkatan sebanyak 70%.

Kata kunci: *high-utility itemset, interestingness,* keranjang belanja, *minimum utility, swarm intelligence.*

IMPROVING THE QUALITY OF HIGH-UTILITY ITEMSETS USING SWARM INTELLIGENCE APPROACH IN THE CASE OF MARKET BASKET ANALYSIS

Ridowati Gunawan [1,*]
[1]*Informatic Study Program, Sanata Dharma University, Indonesia*
**Email : rido@usd.ac.id*

Abstract

There is a paradigm shift in searching for patterns in market basket, initially only paying attention to itemsets that often appear simultaneously, now not only itemsets that often appear but also paying attention to the utility value of the itemsets, known

as high-utility itemsets. The quantity and weight of each item are the main factors to get the high-utility itemset. However, the search space for the high-utility itemset is very wide. Therefore, various techniques are needed to obtain the high-utility itemset so that the results obtained are optimal. The combination of the high-utility itemset and swarm intelligence methods is used to improve the quality of the high-utility itemset. The swarm intelligence approach used is particle swarm optimization which has four main parameters, namely initial population, inertia weight, coefficient acceleration and velocity clamping. Measuring the quality of the high-utility itemset in terms of the number of itemset, total utility, running time, and memory usage. The results of the experiments conducted on market basket dataset showed that the total utility had increased and the factor that most affected the search quality of the high-utility itemset using the swarm intelligence approach was inertia weight. In addition, the use of search without entering the minimum utility value can also improve the quality of high utility itemset, which has increased by 70%.

Keywords: high-utility itemset, interestingness, market basket, minimum utility, swarm intelligence.

Pendahuluan

Analisis terhadap keranjang belanja banyak dilakukan terutama untuk memperoleh pengetahuan tentang kecenderungan belanja dari pelanggan. Pola yang menarik, pola yang tidak terduga dalam keranjang belanja sangatlah penting untuk diperoleh, karena dapat membantu pengambil keputusan untuk menentukan strategi selanjutnya.

Salah satu analisis keranjang belanja adalah mendapatkan asosiasi antar *item*. Pencarian aturan asosiasi diawali proses pencarian *frequent itemset* yaitu *item-item* yang sering muncul secara bersama-sama (Agrawal et al., 1993). Sebuah *itemset* dikatakan *frequent* jika memiliki nilai *support* yang lebih besar dari *minimum support* (min_sup) yang ditetapkan oleh pengguna. *Support* merupakan jumlah transaksi yang terjadi, dimana transaksi merupakan *subset* dari *itemset.* Pencarian *frequent itemset* yang mempertimbangkan jumlah kejadian *itemset* dalam transaksi, yang sering terjadi menjadi *itemset* yang *frequent.*

Pada kasus keranjang belanja, pencarian *itemset* yang menarik tidak hanya mempertimbangkan jumlah kejadian. Nilai kuantitas dan bobot merupakan faktor yang harus dipertimbangkan pula. Bobot dalam hal ini dapat dianalogikan sebagai profit. Pola yang inginkan diperoleh tentunya yang memiliki nilai *profit* yang tinggi dan sekaligus kerap terjadi (*frequent*). Teknik pencarian *itemset* yang mempertimbangkan nilai kuantitas dan nilai dikenal dengan nama *high-utility itemset mining* (HUIM) (Fournier-Viger et al., 2014; Zida et al., 2015). Pencarian tidak lagi berdasarkan nilai *support* tetapi menggunakan pengukuran *utility*. *Itemset* dikatakan memiliki *high-utility* jika memiliki nilai *utility* lebih besar dari nilai *minimum utility* yang ditetapkan oleh penggunanya. *Utility* item merupakan perkalian antara nilai *internal utility* (kuantitas item dalam transaksi) dengan *external utility* (bobot/profit setiap item). Penggunaan kuantitas dan bobot membawa dampak signifikan bagi pengambil keputusan (Y. Liu et al., 2005a), misalnya untuk mengetahui produk yang memberikan kontribusi pada pendapatan

perusahaan dan melihat kecenderungan pelanggan dalam membeli produk (Mai et al., 2017).

Teknik pencarian *high-utility itemset* tidak dapat didekati seperti pada pencarian *frequent itemset*. Pencarian pola yang sering muncul atau *frequent itemset mining* menggunakan *downward closure property* (Agrawal & Srikant, 1994), yang memastikan bahwa subset dari *frequent itemset* adalah *frequent itemset*. Akan tetapi pada *high-utility itemset*, jika sebuah *itemset* memiliki *high utility* maka subset dari *itemset* tersebut belum tentu *high-utility itemset* (Y. Liu et al., 2005b).

Pencarian *high utility itemset* dapat dikelompokan menjadi dua pendekatan yaitu menggunakan konsep pemrograman tradisional dan pendekatan kecerdasan komputasional. Pada umumnya pendekatan tradisional menggunakan algoritma deterministik dan dilakukan berdasarkan pada pencarian *frequent itemset*, seperti Apriori, Eclat dan FP-Growth. Teknik pencarian *high-utility itemset* pendekatan Apriori memiliki ruang pencarian yang sangat besar (Yao et al., 2004a). Penggunaan *two phase algorithm* dilakukan untuk mempercepat proses pencarian melalui konsep *transaction-weighted utilization mining*, hanya kombinasi *itemset* yang memiliki *high transaction weighted utilization* yang ditambahkan sebagai kandidat dalam pencarian *high-utility itemset.* Model ini mempertahankan properti *transaction-weighted downward closure (twdc)* (Y. Liu et al., 2005b). Pendekatan FP-Growth yang dikemukan oleh Han et al. (2000) diterapkan pula dalam pencarian *high-utiity itemset* (Erwin et al., 2007; C.-W. Lin et al., 2011; Tseng et al., 2010). Pendekatan menggunakan struktur list dan tanpa pembentukan kandidat *itemset* memperlihatkan hasil yang lebih baik dibandingkan penelitian sebelumnya (M. Liu & Qu, 2012). Zhang et al. (2018) dalam studi empirisnya menyimpulkan bahwa pemilihan algoritma *high-utility itemset* yang tepat ditentukan dari kepadatan dan panjang *dataset*. Kepadatan merupakan perbandingan rata-rata jumlah *item* dalam keseluruhan transaksi dengan jumlah *item* yang berbeda. Sedangkan panjang *item* merupakan jumlah *item* yang berbeda pada keseluruhan transaksi.

Pendekatan kedua untuk mendapatkan *high-utility itemset* adalah menggunakan paradigma dari kecerdasan komputasional. Menurut Engelbrecht (2007), terdapat lima paradigma kecerdasan komputasional yaitu *artificial neural network (ANN), evolutionary computational (EC), swarm intelligence (SI), artificial immune system (AIS)* dan *fuzzy system (FS).* Tidak semua paradigma sudah dimanfaatkan untuk mendapatkan *high-utility itemset*. Penggunaan algoritma genetika (yang merupakan bagian dari EC) dan SI saja yang telah dicoba untuk mendapatkan *high-utility itemset.*

Kannimuthu & Premalatha (2013) mengusulkan penggunaan algoritma genetika dan fungsi *fitness* yang digunakan adalah nilai *utility* (Yao et al., 2004b). Hasil penelitian menunjukan bahwa waktu eksekusi dan penggunaan memori lebih baik dibandingkan dengan pencarian *high utility itemset* tanpa algoritma genetika. Selain menggunakan nilai *minimum utility* sebagai parameter untuk menyeleksi *itemset* yang menggunakan property `twdc`, juga menggunakan konsep *top-K utility itemset* dari populasi hasil proses evolusi sehingga tidak membutuhkan nilai *minimum utility.*

J. C.-W. Lin et al. (2015) melakukan pencarian *high-utility itemset* menggunakan salah satu pendekatan *swarm intelligence* yaitu *particle swarm*

optimization (PSO). Initial partikel menggunakan *roulette* dan penentuan *velocity* dilakukan secara random. Sementara fungsi yang digunankan unutk melakukan *update velocity* adalah *sigmoid.* menggunakan Tahap awal *itemset* dikodekan secara random. *Itemset* dalam pencarian menggunakan PSO dianalogikan sebagai *particle.* Pengkodean yang diusulkan menggunakan *binary particle swarm optimization (BPSO)*. Hasil evaluasi menunjukan pendekatan menggunakan PSO lebih cepat dibandingkan dengan penggunaan algoritma genetika. Walaupun telah berhasil lebih cepat, parameter untuk BPSO masih menggunakan pengukuran standar yang umum digunakan, yaitu populasi awal dibuat random, *inertia weight* menggunakan padahal pencarian untuk *high-utility itemset* memiliki kekhususan yaitu masukannya berupa rangkaian *itemset* bukan merupakan suatu formula.

Perbaikan algoritma BPSO dilakukan hanya pada proses pengkodean partikel saja yang semula menggunakan *roulette* diubah menjadi menggunakan *OR/NOR tree* (J. C. W. Lin et al., 2017). Nilai dari empat parameter utama dari algoritma BPSO yaitu populasi awal, *inertia weight*, koefisien percepatan dan *velocity clamping* tidak dibahas pada penelitian tersebut. Nilai parameter menggunakan nilai *default* yang digunakan pada algoritma BPSO.

Performa dari PSO dapat ditingkat melalui posisi awal *particle* (populasi awal) (Richards & Ventura, 2004) dan inisialisasi awal merupakan agar algoritma dapat segera mendapatkan nilai optimum (Kazimipour et al., 2014). *Inertia weight*, koefisien percepatan dan *velocity clamping* merupakan parameter yang mempengaruhi proses pembaharuan *velocity*. *Velocity clamping* juga dapat digunakan untuk meningkatkan unjuk kerja algoritma dari BPSO (Alhussein & Haider, 2015).

Pencarian *high utility itemset* tanpa menentukan batas *utility* menggunakan pendekatan BPSO memberikan hasil yang lebih baik ditinjau dari kecepatan, penggunaan memori maupun kualitas *itemset* yang diperoleh dibandingkan dengan yang menggunakan masukan dari pengguna.. Pengukuran kualitas *itemset* diusulkan tidak hanya menggunakan *utility* tetapi juga menggunakan *utility lift, utility confidence* (Gunawan et al., 2020).

Penelitian menggunakan pendekatan *swarm intelligence* pada umumnya diaplikasikan pada data publik yang umum digunakan untuk menguji algoritma *firequent itemset mining* (http://fimi.ua.ac.be/data). Sehingga pembentukan nilai *utility* dilakukan dengan cara dibangkitkan secara random. Tidak menunjukan kondisi yang sesungguhnya.

Penelitian ini akan menerapkan algoritma pencarian *high utility itemset* menggunakan pendekatan *swarm intelligence* yaitu PSO pada data yang diperoleh dari transaksi keranjang belanja. Nilai *utility* atau nilai profit merupakan nilai yang sesungguhnya. Selain menggunakan data sesungguhnya, penelitian ini mencoba untuk mencari, dari keempat parameter PSO, parameter apa yang paling mempengaruhi kinerja algoritma PSO sehingga mampu untuk meningkatkan kualitas dari *high-utility itemset* yang diperoleh untuk kasus pada keranjang belanja. Analisis *high-utility itemset* ditinjau dari banyaknya *itemset* yang diperoleh, total *utility* dari *itemset* yang diperoleh, kecepatan akses, memori yang digunakan. Nilai *minimal utility* tidak ditetapkan oleh pengguna tetapi menggunakan algoritma yang diusulkan oleh (Gunawan et al., 2020). Sebagai pembanding usulan parameter akan dibandingkan dengan algoritma BPSO menggunakan parameter standar dari J. C. W. Lin et al. (2017).

Pada bagian selanjutnya akan dibahas tentang metode penelitian yang digunakan, hasil dan pembahasan dan diakhiri dengan penutup yang berisi kesimpulan dan saran untuk pengembangan penelitian selanjutannya.

Metode Penelitian

Pada bagian ini dijelaskan tentang gambaran umum penelitian, Langkah-langkah penelitian serta bahan penelitian.

Gambaran Umum Penelitian

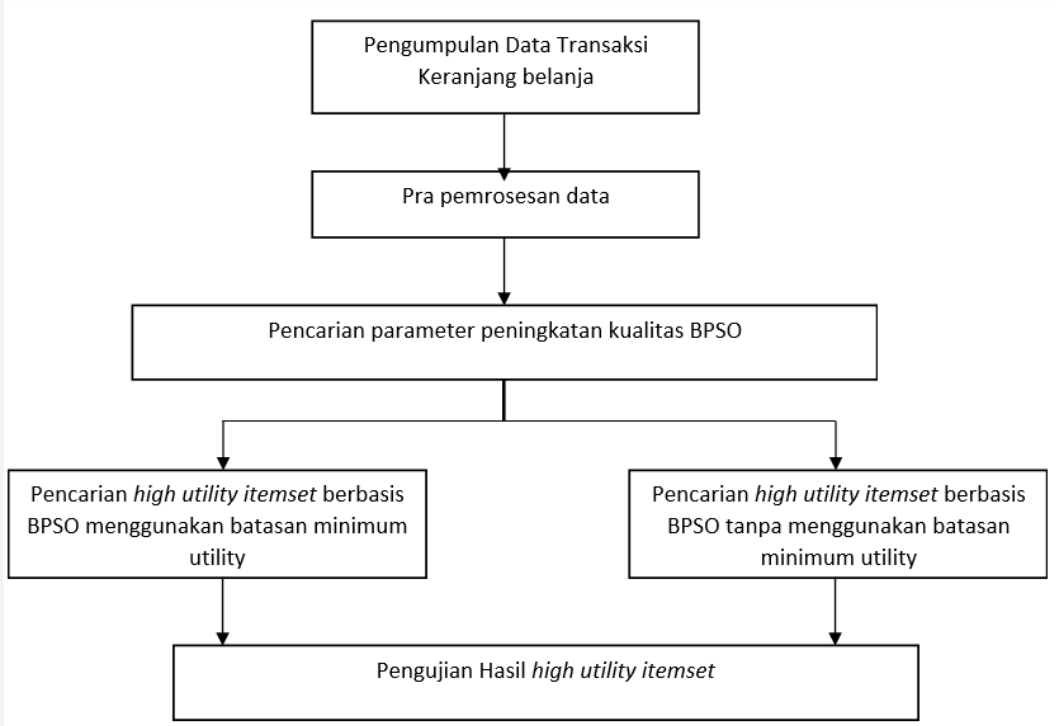
Sistem mengolah data transaksi dari keranjang belanja, dalam setiap transaksi berisi informasi item-item dan juga profit dari setiap *item.* Selanjutnya data akan diolah untuk mendapatkan *high utility itemset* menggunakan pendekatan *swarm intelligence.* Pendekatan *swarm intelligence* yang digunakan adalah *particle swarm optimization*. Pada proses pencarian ini terlebih dahulu akan dilakukan percobaan parameter apa yang paling optimum untuk mendapatkan *high utility itemset.*

Setelah proses mendapatkan *high-utility itemset*, langkah selanjutnya adalah melakukan proses pengujian terhadap model yang dibangun, ditinjau dari jumlah *itemset* yang diperoleh, total *utility* yang diperoleh, waktu eksekusi dan memori yang digunakan.

Secara garis besar penelitian ini mengembangkan model untuk mendapatkan konfigurasi *BPSO* yang tepat untuk mendapatkan *high-utility itemset* dengan menggunakan parameter seleksi yaitu *minimum utility* dan tanpa menggunakan parameter seleksi.

Langkah Penelitian

Langkah penelitian untuk mendapatkan *high utility itemset* menggunakan pendekatan *swarm intelligence* dapat dilihat pada Gambar 1.



Gambar 1. Langkah Penelitian

Langkah penelitian sebagai berikut:

1. Pengumpulan data transaksi keranjang belanja. Transaksi keranjang belanja yang digunakan adalah transaksi penjualan pada bulan Januari 2022. Transaksi penjualan yang dikumpulkan dalam bentuk tabel transaksi yang terdiri dari tabel penjualan, detail Penjualan, master barang.
2. Pra pemrosesan data. Prapemrosesan data penjualan dibagi dua proses utama yaitu proses menggabungkan data penjualan dan detail penjualan untuk mendapatkan informasi transaksi yang berisi data barang beserta jumlah barang yang dijual dan proses untuk mendapatkan nilai profit barang. Data tabel penjualan dan detail Penjualan diubah bentuk ke dalam list. Setiap transaksi memiliki TID dan itemset dan informasi jumlah barang. Proses mendapatkan nilai profit dari setiap barang diperoleh dari pengurangan harga jual dengan harga beli. Untuk mengurangi waktu komputasi dilakukan proses pengubahan kode barang menjadi angka yang berurutan mulai dari angka 1.
3. Pencarian parameter peningkatan kualitas BPSO. Sebelum melakukan proses pencarian *high-utility itemset* terlebih dahulu ditentukan parameter-parameter yang mempengaruhi proses pencariannya. Karena proses pencarian menggunakan pendekatan BPSO. maka nilai parameter akan mengikuti parameter dari BPSO. Paramater yang harus ditentukan adalah populasi awal, *inertia weight, coefficient acceleration*, dan *velocity clamping.*
    a. Populasi awal dapat menggunakan *zero vector, frequent itemset* atau sejumlah transaksi yang memiliki total utility tertinggi.
    b. *Inertia weight* yang dapat mempengaruhi adalah *Increasing Inertia Weight* (J. Liu et al., 2016), *adaptive inertia weight* (Nickabadi et al., 2011). *Constant inertia weight with probabilistic velocity* (Tasgetiren & Liang, 2004), *constriction factor* (Clerc, 1999), dan bernilai *static* tidak dipengaruhi oleh perubahan *velocity* biasanya ditetapkan 0.9.
    c. *Coefficient acceleration*. Koefisien percepatan terdiri dari dua yaitu faktor individu dan faktor sosial. Keduanya akan ditetapkan sama yaitu 2.
    d. *Velocity clamping*. Metode *velocity clamping* agar tidak terjadi *overfitting.* Metode ini menetapkan batas kecepatan maksimal dari partikel atau *velocity maximum*, yang digabungkan fungsi *piece-wise linear* (Tasgetiren & Liang, 2004).
4. Pencarian *high-utility itemset* berbasis BPSO dengan batasan *minimum utility*. Nilai batas *minimum utility* ditetapkan dalam ukuran persentase terhadap total utility dalam transaksi. Ditetapkan nilainya dari 0,1 % sampai dengan 5%.
5. Pencarian *high-utility itemset* berbasis BPSO tanpa batasan *minimum utility*. Pada bagian ini akan menggunakan hasil parameter terbaik pada tahapan sebelumnya dan menggunakan alur berpikir seperti yang telah dikemukakan oleh (Gunawan et al., 2020).

6. Pengujian hasil *high-utility itemset.* Hasil ditinjau dari waktu eksekusi, memori yang dibutuhkan serta jumlah *itemset dan total utility* yang diperoleh. Skenario pengujian yang dilakukan adalah :
    a. Membandingkan pencarian *high utility itemset* menggunakan parameter BPSO standar dengan *high utility itemset* menggunakan parameter yang telah diperbaharui menggunakan batas minimum *utility*.
    b. Membandingkan pencarian *high utility itemset* menggunakan parameter BPSO yang telah diperbahuri dengan menggunakan *minimum utility* dan tanpa *minimum utility.*

Bahan Penelitian

Bahan yang digunakan dalam penelitian ini adalah :

1. Transaksi dari keranjang belanja yang diperoleh dari transaksi sesungguhnya dari perusahaan ritel. Data diperoleh dari transaksi selama bulan Januari 2022.
2. Pustaka program *third-party* yang tersedia yait SPMF: Java Open-Source Pattern Mining Library (Fournier-Viger et al., 2016).
3. Pengolah data menggunakan *database management system MySQL* menggunakan IDE SQLYog.
4. Implementasi algoritma menggunakan Bahasa pemrograman Java (versi Jdk 1.8.0_72) dan mengguanakan IDE Netbeans 8.2
5. Spesifikasi peralatan adalah PC dengan AMD Ryzen 7 2700 Eight-Core Processor, 16 GB RAM menggunakan 64 bit Windows 10.

Hasil dan Pembahasan

Bagian ini akan dibahas hasil implementasi dari langkah-langkah penelitian serta hasil yang diperoleh selama penelitian.

Hasil Pengumpulan Data

Data terdiri dari tiga buah tabel yaitu tabel Penjualan, detail Penjualan dan master barang. Keterangan tentang ketiga tabel beserta jumlah data dapat dilihat pada Tabel 1.

Table 1. Karakteristik Kumpulan Data

Nama Tabel	Keterangan	Jumlah Record
Penjualan	Berisi informasi header penjualan yaitu noFaktur, tglFaktur, kodePelanggan, totalPenjualan	1.898
Detail Penjualan	Berisi informasi detail penjualan terdiri dari noFaktur,kodebarang,namabarang, quantity, satuan, hargasatuan, hpperbarang,	7.461
Master Barang	Berisi informasi detail tentang barang yang terdiri dari field kodebarnag, namabarang, kategori, ukuran, harga beli, harga jual	4.071

Contoh tabel detail penjualan yang merupakan hasil relasi antara tabel detail penjulaan dan master barang dilihat pada Tabel 2. Pada tabel 2 terdiri dari lima buah transaksi, masing-masing transaksi merekam beberapa barang yang berbeda-beda. Sebagai contoh, transaksi ke-4 merekam 2 jenis barang.

Table 2. Contoh Data Detail Penjualan

TID	No Faktur	Kode Barang	Nama Barang	Quantity	Satuan	Harga Satuan	Hppperbarang
1	SG1220103001	PBS15KGDS	PALMIA BOS 15 KG	2	CRT	534700	456798
2	SG1220103002	JAVADARK5	OH JAVA DARK 5 KG	3	ZAK	400000	319905
	SG1220103002	INKOPPASLT	IN KOPYOR PASTA 1LT	5	PCS	125000	76182
3	SG1220103003	DIBKRSTARBR	BAKERSTAR BIRU 25 KG	4	ZAK	195000	162000
4	SG1220103004	INREDVELVET	IN POINT CAFE DRINKING PWD RED VELVET 1 KG	3	KG	70000	34545
	SG1220103004	GRENTEMIX	IN CHATRAMUE BRAND GREEN TEA MIX 200 GR	5	PCS	62500	4230
5	SG1220103005	GMKCCFL4X5	GMK CHOCO FILLING-4X5KG	5	CRT	694000	539427
	SG1220103005	MZMTXX25	MAIZENA MATAHARI 25 KG	1	ZAK	340000	245000
	SG1220103005	AGNCRTR1KG	CREAM OF TAR TAR 1KG/10	3	CRT	545000	44909
	SG1220103005	PKPSTRWHIT	COLATTA PASTRY COMP. WHITE 12X1KG	4	CRT	516000	417960

Hasil Prapemrosesan Data

Tahapan ini digunakan untuk mendapatkan data yang siap untuk diproses, karena data yang dikumpulkan terdiri dari beberapa tabel maka proses yang dilakukan adalah menggabungkan ketiga tabel. Tahapan pertama adalah menggabungkan data penjualan dengan data detail penjualan, hasil seperti pada Tabel 2. Kemudian ditambahkan dua kolom baru yaitu *index kode* yang merupakan index dari kodebarang agar lebih cepat dalam melakukan proses serta kolom *utility* yang merupakan profit yang diperoleh dari kolom hargasatuan dikurangkan dengan hppperbarang. Selanjutnya setiap transaksi dibuat dalam satu buah transaksi yang berisi informasi nofaktur, list kode (kumpulan index kode dari setiap faktur) dan utility yang juga berupa list utility yang bersesuain dengan list kode. Hasil dari lima transaksi pada Tabel 2 dapat dilihat pada Tabel 3. Pada transaksi ke-4 yang terdiri

dari 2 buah barang pada kolom indexkode menjadi 365 dan 558, pada kolom utility berisi 55937 merupakan utility dari 365 dan 44373 merupakan utility dari 558.

Table 3. Faktur Utility

No Faktur	IndexKode	Utility
SG1220103001	764	79890
SG1220103002	628 484	84109 48818
SG1220103003	180	52673
SG1220103004	365 558	55937 44373
SG1220103005	781 37 725 300	90300 293646 164314 96740

Selanjutnya data disimpan dalam format csv dengan format setiap baris adalah <indexkode>:<total utility>:<utility>. Kelima data pada Tabel 3, diubah menjadi format seperti pada Tabel 4, daftar transaksi.

Table 4. Daftar Transaksi

Transaksi <kode>:<totalutility>:<utility>
764:79890:79890
628 484:132927:84109 48818
180:52673:52673
365 558:100310:55937 44373
781 37 725 300:645000:90300 293646 164314 96740

Pencarian Parameter BPSO

Untuk mendapatkan parameter terbaik BPSO dilakukan percobaan kombinasi dari keempat parameter utama dari BPSO, yaitu melakukan kombinasi populasi awal dan *inertia weight.* Sementara untuk nilai *coefficient acceleration* ditetapkan 2 dan nilai *velocity clamping* dibuat beberapa variasi nilai. Konfigurasi terbaik yang diperoleh adalah penggunaan *inertia weight* menggunakan *adaptive inertia weight* dan *constriction factor,* sementara untuk populasi awal tidak banyak mempengaruhi hasil performa algoritma BPSO (Gunawan et al., 2022).

Berdasarkan hasil tersebut, maka untuk proses pencarian *high utility itemset* berbasis *swarm intelligence* ini menggunakan konfigurasi parameter BPSO sebagai berikut:

1. Populasi awal menggunakan kumpulan dari *frequent itemset*, diambil 20 itemset yang memiliki nilai *support* tertinggi.
2. *Inertia weight* menggunakan rumusan *adaptive inertia weight.* Untuk berbagai kasus menunjukan kinerja yang baik juga pada pencarian *high utility itemset* berbasis PSO (Nickabadi et al., 2011) dan terbukti juga dalam beberapa percobaan cukup baik untuk BPSO.
3. *Coefficient acceleration* menggunakan faktor individu c1 dan faktor sosial c2 yang sama yaitu 2.
4. *Velocity clamping* selalu diterapkan bersamaan dengan *piece-wise linear* dan nilai batas yang diterapkan adalah [-4, 4]. Batas kecepatan maksimum partikel adalah 4 (Vmax= 4) dan batas kecepatan minimum adalah -4 (Vmin=-4).

Algoritma untuk memdapatkan parameter terbaik menggunakan modifikasi dari algoritma standar yang diusulkan oleh J. C. W. Lin et al. (2017)

Hasil Pencarian High Utility Itemset Menggunakan Parameter Terbaik

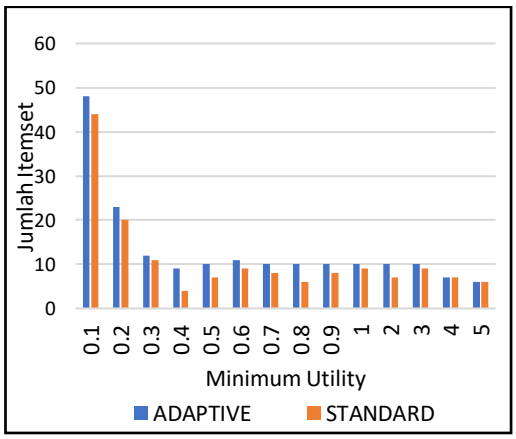
Proses pencarian *high utility itemset* berbasis BPSO terdiri dari 4 proses utama yaitu (1) prapemrosesan untuk menyimpan informasi item-item beserta *utility* dari setiap *item* serta total utility dari setiap transaksi, (2) pengkodean partikel yaitu mengkodekan partikel dengan nilai biner 0 atau 1, (3) evaluasi fungsi *fitness*, fungsi *fitness* yang digunakan adalah nilai *utility* dan (4) proses perubahan partikel, pada bagian ini akan menggunakan fungsi Sigmoid dengan fungsi *inertia weight* menggunakan *adaptive inertia weight.*

Tabel perbandingan parameter BPSO standar dengan parameter BPSO yang diusulkan dapat dilihat pada Tabel 5.

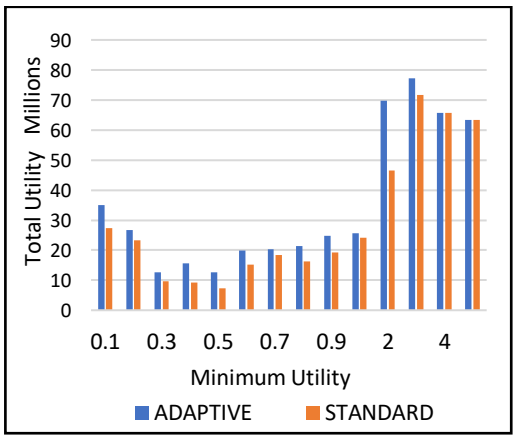
Tabel 5. Karakteristik Perbandingan Parameter BPSO

Parameter	BPSO Adaptive (Usulan)	BPSO Standard (J. C. W. Lin et al., 2017)
Populasi awal	Frequent Itemset	Vector 0
Inertia weight	Adaptive Inertia Weight Wmax=0.2; Wmin=1	Static inertia weight =0.9
Coefficient Accelaration	C1 = 2; C2 = 2	C1 = 2; C2=2
Velocity Clamping	Menggunakan *piece-wise linear* Vmin = -4 Vmax= 4	Tidak menggunakan *piece-wise linear* Vmin = -2 Vmax = 2

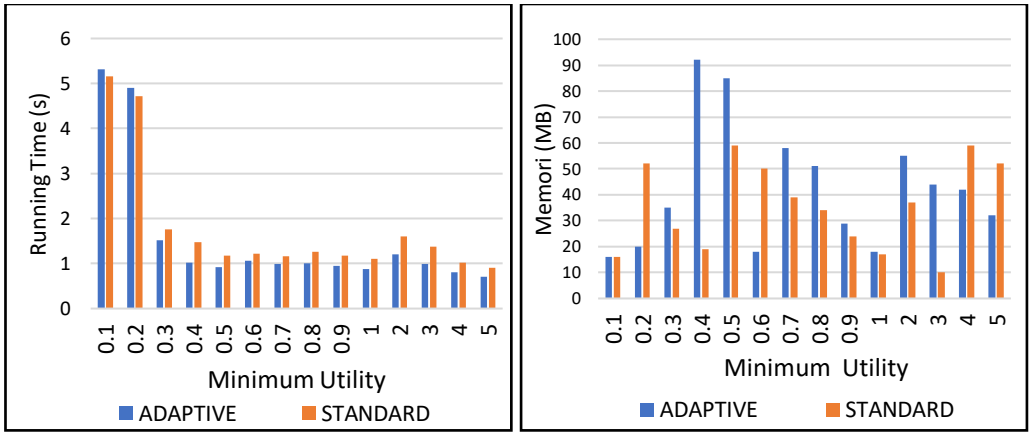
Gambar 2 merupakan tabel hasil perbandingan pencarian *high utility itemset* menggunakan parameter *standard* dan parameter *adaptive* untuk berbagai nilai *minimum utility*. Jumlah populasi yang digunakan adalah 20 dan jumlah iterasi 1000. Setiap nilai *minimum utility* akan dilakukan percobaan sebanyak 3 kali dan hasil setiap pengukuran akan diambil rata-rata. Baris pertama adalah grafik perbandingan jumlah *itemset* yang diperoleh (a) dan total *utility* (b) , baris kedua perbandingan *runtime* (c) dan terakhir penggunaan memori (d)



(a)



(b)



(c) (d)

Gambar 2. Perbandingan jumlah *itemset*, total *utility* (baris pertama), *running time* dan penggunaan memori (baris kedua) dari metode BPSO-*standard* dan BPSO-*adaptive* terhadap berbagai nilai *minimum utility*

Jumlah item dan total *utility* BPSO-*adaptive* lebih banyak dibandingkan BPSO-*standard*. *Running time* BPSO-*adaptive* lebih cepat, akan tetapi untuk penggunaan memori tidak dapat disimpulkan. Penggunaan memori secara rata-rata BPSO-*adaptive* membutuhkan memori lebih banyak. Secara umum penggunaan *adaptive inertia* sebagai parameter BPSO lebih baik dibandingkan yang menggunakan *static inertia.*

```
1. #ITEM: 72 #UTIL: 491640
2. #ITEM: 119 #UTIL: 1954044
3. #ITEM: 917 #UTIL: 395507
4. #ITEM: 97 #UTIL: 456952
5. #ITEM: 359 #UTIL: 393016
6. #ITEM: 858 #UTIL: 6593100
7. #ITEM: 684 #UTIL: 1311408
8. #ITEM: 361 #UTIL: 710792
9. #ITEM: 45 97 262 #UTIL: 394296
10. #ITEM: 180 #UTIL: 684749
11. #ITEM: 97 262 #UTIL: 393356
12. #ITEM: 74 #UTIL: 446395
13. #ITEM: 680 #UTIL: 2034215
--------------------------
Total Utility :16259470
MinUtility : 383165
InitialUtililty: 0.003
Total time ~ 1518 ms
Memory ~ 35.4677734375 MB
High-utility itemsets count : 13
=========================================
```

Gambar 3. Hasil *High Utility Itemset* Adaptive dengan *Minimum Utility 0.3%*

Contoh hasil program untuk *minimum utility = 0.2 %* menggunakan *adaptive* terlihat pada Gambar 3. Contoh hasil baris ke-9 *itemset* yang diperoleh adalah *item* 45, 97, 262 dan total *utility* adalah 394296 rupiah. Item 45, 97, 262 berturut-turut

adalah PASTA VANILLA (25 ML), INDIAN BLACK BROWN RAISIN 10KG LOS,INDIAN BLACK BROWN RAISIN 10KG LOS.

Hasil Pencarian High Utility Itemset Tanpa Minimum Utility

Pencarian *high utility itemset* dengan menetapkan nilai *minimum utility* sangatlah menyulitkan bagi pengguna. Pengguna tidak mengetahui dengan pasti berapa nilai yang tepat *minimum utility* yang digunakan. Oleh karena usulan pemanfaatan algoritma pencarian *high utility itemset* sangatlah bermanfaat. Penggunaa hanya perlu memberikan masukan kepada sistem berapa *high utility itemset* yang diinginkan. Jika penggunaa hanya ingin meninjau 100 *high utility itemset* maka sistem akan dapat memberikan hasil sebanyak 100 *high utility itemset* yang telah diurutkan berdasarkan total *utility* nya.

Pencarian tanpa penentuan nilai *minimum utility* dilakukan karena algoritma berbasis *swarm intelligence* selalu mencari nilai optimum sesuai fungsi *fitness* nya sehingga tidak perlu batasan nilai *utility*. Pembatasan pencarian hanya cukup dari banyaknya iterasi yang dikehendaki. Makin banyak iterasi yang dilakukan maka akan semakin membutuhkan waktu yang lama. Performa algoritma ini dilihat dari jumlah *itemset* yang diperoleh dan total utility yang diperoleh untuk sejumlah *k itemset* yang diinginkan.

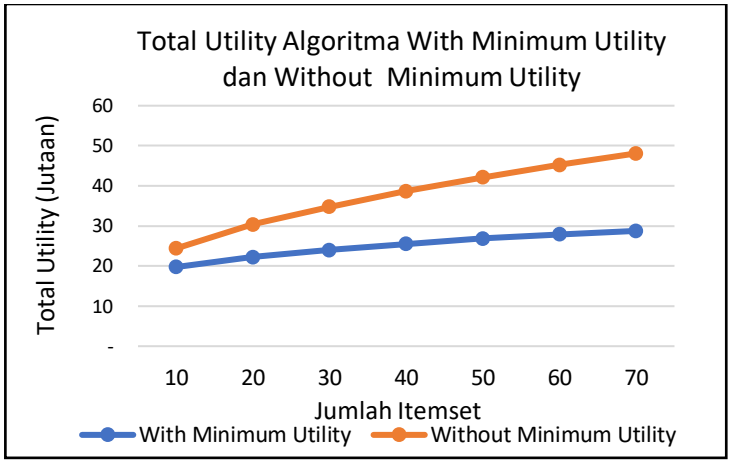
Untuk melihat performa algoritma yang digunakan, dilakukan percobaan untuk mendapatkan *high utility itemset* menggunakan ukuran populasi 10, 20 dan 30 dan jumlah iterasi sebanyak 1000 dan 2000 menggunakan konfigurasi BPSO-adaptive. Hasil percobaan untuk konfigurasi tersebut dapat dilihat pada Tabel 6.

Tabel 6. Hasil BPSO Adaptive Tanpa Minimum Utillity

Jumlah Iterasi	Ukuran Populasi	Jumlah Item	Total Utility (100)	Waktu (s)	Memory (MB)
1000	10	478	51053712	7	24
	20	628	56592506	14	43
	30	688	58298646	22	26
2000	10	576	52091566	14	21
	20	650	52920074	28	33
	30	744	55140590	42	26

Hasil percobaan memperlihatkan bahwa semakin besar jumlah iterasi maka jumlah item dan total *utility* untuk 100 *itemset* menghasilkan hasil yang lebih baik. Kecepatan waktu semakin tinggi sedangkan memori tidak dapat disimpulkan. Hal yang sama terjadi Ketika jumlah ukuran populasi ditingkatkan.

Selanjutnya hasil perbandingan antara pencarian *high utility itemset* menggunakan masukan *minimum utility* dan tanpa menggunakan *minimum utility*. Keduanya akan menggunakan pendekatan BPSO adaptive sebagai pendekatan yang lebih baik dari pendekatan BPSO standard. Jumlah total *utility* untuk 10, 20, 30, 40, 50, 60, 70 *itemset* akan dihitung. Untuk BPSO adaptive agar dapat menghasilkan jumlah *itemset* lebih dari 100 menggunakan nilai *minimum utility = 0,05%*. Ukuran populasi = 20 dan banyaknya iterasi adalah 1000. Hasil percobaan berupa grafik perbandingan dapat dilihat pada Gambar 4.



Gambar 4. Perbandingan Total Utility Algoritma BPSO with Minimum Utility dan BPSO without Minimum Utility

Hasil percobaan memperlihatkan bahwa penggunaan algoritma BPSO *adaptive* tanpa batasan *minimum utility* memberikan hasil yang lebih baik dibandingkan menggunakan *minimum utility*. Rata-rata peningkatan total *utility* yang diperoleh sebesar 70%.

Kesimpulan

Berdasarkan penelitian yang dilakukan terbukti bahwa *inertia weight* merupakan parameter utama dalam peningkatan kinerja pencarian *high utility itemset* berbasis *swarm intelligence*. Kombinasi parameter BPSO terbaik adalah populasi awal menggunakan *frequent itemset*, *inertia weight* menggunakan *adaptive inertia weight*, nilai *coefficient acceleration* menggunakan nilai 2 dan *velocity clamping* menggunakan *piece-wise*. Terjadi peningkatan performa hasil pencarian *high utility itemset* menggunakan BPSO *adaptive* ketika tanpa memasukan nilai minimum utility ditinjau dari total *utility* yang diperoleh yaitu sebesar 70%.

Saran

Untuk pengembangan selanjutnya, pencarian *high utility itemset* menggunakan pendekatan *swarm intelligence* dapat menggunakan algoritma lain selain *particle swarm optimization* seperti *ant colony*, *bat*, *fish* dan lainnya.

Daftar Pustaka

Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, *22*(2), 207–216. https://doi.org/10.1145/170036.170072

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. *Proceedings of the 20th International Conference on Very Large Data Bases*, *1215*, 487–499. https://doi.org/10.1.1.40.6757

Alhussein, M., & Haider, S. I. (2015). Improved Particle Swarm Optimization Based on Velocity Clamping and Particle Penalization. *2015 3rd International*

Conference on Artificial Intelligence, Modelling and Simulation (AIMS), 61-64. https://doi.org/10.1109/AIMS.2015.20

Clerc, M. (1999). The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. *Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999*, *3*, 1951–1957.

Engelbrecht, A. P. (2007). *Computational Intelligence: An Introduction* (Second). John Wiley & Sons, Ltd. https://doi.org/10.1007/978-3-319-25964-2_2

Erwin, A., Gopalan, R. P., & Achuthan, N. R. (2007). CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach. *7th IEEE International Conference on Computer and Information Technology (CIT 2007)*, 71–76. https://doi.org/10.1109/CIT.2007.120

Fournier-Viger, P., Lin, C. ., Gomaris, A., Gueniche, T., Soltani, A., Deng, Z., & Lam, H. T. (2016). SPMF: a Java Open-Source Pattern Mining Library Version 2. *Proc. 19th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD 2016*, 36–40. http://www.philippe-fournier-viger.com/spmf/

Fournier-Viger, P., Wu, C. W., Zida, S., & Tseng, V. S. (2014). FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8502 LNAI*(June), 83–92. https://doi.org/10.1007/978-3-319-08326-1_9

Gunawan, R., Winarko, E., & Pulungan, R. (2020). A BPSO-based method for high-utility itemset mining without minimum utility threshold. *Knowledge-Based Systems*, *190*. https://doi.org/10.1016/j.knosys.2019.105164

Gunawan, R., Winarko, E., & Pulungan, R. (2022). Performance comparison of inertia weight and acceleration coefficients of BPSO in the context of high-utility itemset mining. *Evolutionary Intelligence*, *0123456789*. https://doi.org/10.1007/s12065-022-00707-0

Han, J., Jian, P., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *SIGMOD ’00 Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, *1*, 1–12.

Kannimuthu, S., & Premalatha, K. (2013). Discovery of high utility itemsets using genetic algorithm. *International Journal of Engineering and Technology*, *5*(6), 4866–4880. https://doi.org/10.1080/08839514.2014.891839

Kazimipour, B., Li, X., & Qin, A. K. (2014). A review of population initialization techniques for evolutionary algorithms. *2014 IEEE Congress on Evolutionary Computation (CEC)*, 2585–2592. https://doi.org/10.1109/CEC.2014.6900618

Lin, C.-W., Hong, T.-P., & Lu, W.-H. (2011). An effective tree structure for mining high utility itemsets. *Expert Systems with Applications*, *38*(6), 7419–7424.

Lin, J. C.-W., Yang, L., Fournier-Viger, P., Frnda, J., Sevcik, L., & Voznak, M. (2015). An Evolutionary Algorithm to Mine High-Utility Itemsets. *Advances in Electrical and Electronic Engineering*, *13*(4), 392–398.

Lin, J. C. W., Yang, L., Fournier-Viger, P., Hong, T. P., & Voznak, M. (2017). A binary PSO approach to mine high-utility itemsets. *Soft Computing*, *21*(17), 5103–5121. https://doi.org/10.1007/s00500-016-2106-1

Liu, J., Mei, Y., & Li, X. (2016). An analysis of the inertia weight parameter for binary particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *20*(5), 666–681. https://doi.org/10.1109/TEVC.2015.2503422

Liu, M., & Qu, J. (2012). Mining High Utility Itemsets without Candidate Generation Categories and Subject Descriptors. *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM)*, 55–64.

Liu, Y., Liao, W., & Choudhary, A. (2005a). A fast high utility itemsets mining algorithm. *Proceedings of the 1st International Workshop on Utility-Based Data Mining - UBDM ’05*, 90–99. https://doi.org/10.1145/1089827.1089839

Liu, Y., Liao, W., & Choudhary, A. (2005b). A two-phase algorithm for fast discovery of high utility itemsets. *Proceeding PAKDD’05 Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, 689–695. https://doi.org/http://dx.doi.org/10.1007/11430919_79

Mai, T., Vo, B., & Nguyen, L. T. T. (2017). A lattice-based approach for mining high utility association rules. *Information Sciences*, *399*, 81–97. https://doi.org/10.1016/j.ins.2017.02.058

Nickabadi, A., Ebadzadeh, M. M., & Safabakhsh, R. (2011). A novel particle swarm optimization algorithm with adaptive inertia weight. *Applied Soft Computing Journal*, *11*(4), 3658–3670. https://doi.org/10.1016/j.asoc.2011.01.037

Richards, M., & Ventura, D. (2004). Choosing a starting configuration for particle swarm optimization. *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat.No.04CH37541)*, *3*, 2309–2312. https://doi.org/10.1109/IJCNN.2004.1380986

Tasgetiren, M. F., & Liang, Y.-C. (2004). A Binary Particle Swarm Optimization Algorithm for Lot Sizing Problem. *Journal of Economic and Social Research*, *5*(2), 1–20. http://jesr.journal.fatih.edu.tr/jesr.tasgetiren-liang.pdf

Tseng, V., Wu, C., Shie, B., & Yu, P. (2010). {UP-Growth}: an efficient algorithm for high utility itemset mining. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 253–262.

Yao, H., Hamilton, H. J., & Butz, C. J. (2004a). A Foundational Approach to Mining Itemset Utilities from Databases. *Proceedings of the 2004 SIAM International Conference on Data Mining*, 482–486. https://doi.org/10.1137/1.9781611972740.51

Yao, H., Hamilton, H. J., & Butz, C. J. (2004b). A Foundational Approach to Mining Itemset Utilities from Databases. *Proceedings of the 2004 Society for Industrial and Applied Mathematics (SIAM) International Conference on Data Mining*, 482–486.

Zhang, C., Almpanidis, G., Wang, W., & Liu, C. (2018). An empirical evaluation of high utility itemset mining algorithms. *Expert Systems with Applications*, *101*, 91–115.

Zida, S., Fournier-Viger, P., Lin, J. C. W., Wu, C. W., & Tseng, V. S. (2015). EFIM: A highly efficient algorithm for high-utility itemset mining. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9413*, 530–546. https://doi.org/10.1007/978-3-319-27060-9_44