Penerapan Algoritma *Greedy* dalam Menentukan Sampel TPS pada *Quick count*

Dwitri Desvira 13510044[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13510044@std.stei.itb.ac.id

***Abstract*—Algoritma *Greedy* merupakan salah satu algoritma yang dapat menangani masalah optimasi. Makalah membahas bagaimana algoritma *greedy* dapat dimanfaatkan dalam menentukan sampel TPS yang dibutuhkan pada proses perhitungan cepat atau *quick count*. Metode yang digunakan pada makalah ini adalah menggunakan jumlah pemilih pada TPS dengan membandingkannya dengan rata-rata jumlah pemilih pada daerah tersebut.**

Index Terms*—*greedy, quick count, pemilu, tps.

I. PENDAHULUAN

Pemilihan umum anggota legislatif ataupun pemilihan umum presidan merupakan pesta rakyat terbesar yang diadakan sekali dalam lima tahun. Pemilihan umum yang akrab disebut Pemilu Legislatif ini bertujuan untuk memilih anggota DPR, DPD, serta DPRD se Indonesia, yang akan menjabat dalam periode waktu tertentu. Selain pemilu yang serentak diadakan di Indonesia, terdapat juga pemilihan umum yang diadakan untuk memilih pejabat daerah, seperti Pilkada. Pilkada sendiri diselenggarakan oleh Komisi Pemilihan Umum (KPU) Provinsi dan KPU Kabupaten/Kota dengan diawasi oleh Panitia Pengawas Pemilihan Umum (Panwaslu) Provinsi dan Panwaslu Kabupaten/Kota.

Pemungutan suara pada pemilu legislatif atau pada pemilu presiden diadakan serentak di seluruh Indonesia. Setelah proses pemungutan suara selesai, diadakan proses penghitungan suara diseruluh TPS di Indonesia. Rekapitulasi resmi yang dilakukan oleh KPU biasanya memakan waktu hingga tiga minggu. Hal tersebut disebabkan oleh banyaknya data yang harus dikumpulkan dari seluruh wilayah di Indonesia.

Melihat lamanya hasil rekapitulasi resmi yang dilakukan oleh KPU, diadakanlah proses perhitungan cepat untuk mendapatkan hasil sementara dari pemilu yang telah diselenggarakan. Proses perhitungan cepat inilah yang biasa disebut sebagai *Quick count*. *quick count* adalah metode hitung cepat yang berbasis pada pengambilan sampel yang digunakan untuk menunjukkan hasil sementara dengan tingkat kesalahan (error) yang sangat kecil. Dengan demikian hasil hitung cepat tidak akan berbeda jauh dengan perhitungan secara manual oleh KPU. Untuk lebih jelasnya, gambar dibawah ini berisi perbedaan perhitungan yang dilakukan secara resmi dengan perhitungan *Quick count*.

	KPU/KPUD	Lembaga Quick Count
Jumlah TPS	Semua TPS	TPS yang dipilih secara acak
Petugas	Petugas KPU/KPUD	Relawan
Parameter	Berdasarkan rekapan data yang telah dihitung dari TPS, lalu Kelurahan, Kecamatan, Kabupaten, Propinsi dan Pusat	Berdasarkan data yang diinput oleh relawan yang dikirim ke server pusat.
Waktu	Beberapa hari/minggu	2-3 jam setelah perhitungan suara selesai

Gambar 1. Perbedaan perhitungan surat suara [Sumber : http://www.kaskus.co.id/thread/534659cb17410e8b461fquick-count-dalam- pemilu/1]

Lazimnya, *Quick count* dilakukan oleh lembaga atau individu yang memiliki kepentingan terhadap proses dan hasil pemilu. Tujuan dan manfaat dari *quick count* adalah agar pihak-pihak yang berkepentingan memiliki data pembanding yang dapat digunakan untuk mendeteksi adanya kemungkinan kecurangan yang terjadi pada proses tabulasi suara.

Quick count berbeda halnya dengan survey. Akurasi hasil *quick count* lebih akurat, hal ini karena *quick count* menghitung hasil pemilu langsung dari TPS target, bukan berdasarkan opini dari responden. Oleh karena itu, untuk mendapatkan akurasi yang tinggi, dibutuhkan metode yang tepat dalam pelaksanaannya. Salah satu parameter keberhasilan pada *quick count* adalah ketepatan dalam menentukan TPS yang akan dijadikan sebagai sampel

II. DASAR TEORI

A. *QUICK COUNT*

Ujiyati, 2004, medefinisikan *quick count* sebagai proses pencatatan hasil perolehan suara di ribuan Tempat Pemungutan Suara (TPS) yang dipilih secara acak. Selain untuk memperkirakan hasil pemilu melalui hasil perhitungan sementara, *Quick count* juga memiliki fungsi sebagai alat control yang mampu mendeteksi penyimpangan dan kecurangan yang terjadi selama proses

penyelenggaraan pemilu.

Menurut Lembaga Survei Indonesia (LSI), *quick count* dibagi menjadi 2 yaitu Real *Quick count* dan Sampling *Quick count*. Real *Quick count* merupakan proses perhitungan cepat dengan menjadikan seluruh TPS sebagai sumber data, untuk Real *Quick count* hasil yang didapat akan menjadi representasi hasil pemilihan secara utuh. Sedangkan Sampling *Quick count* adalah perhitungan cepat yang umum dilakukan dengan menentukan sampel TPS dengan jumlah dan cara tertentu sesuai ilmu statistika, sehingga hasil yang didapatkan akan memiliki bias kesalahan (error) dari hasil seutuhnya.

Masing-masing lembaga yang melakukan *quick count* memiliki metode yang berbeda-beda dalam pengambilan sampelnya. Namun, pada umumnya cara kerja *quick count* yang banyak dilakukan adalah sebagai berikut.

1. Mempersiapkan perangkat serta sistem pendukung untuk bisa memberikan data secara cepat ke pusat pengolah data lembaga survei yang melakukan metode *Quick count* ini. Perangkat ini mulai dari komputer untuk meng-input-kan data hingga ponsel untuk mengirim SMS hasil pemilu ke server tempat menerima data.
2. Pemilihan TPS sebagai tempat pengambilan data. TPS yang di ambil secara acak berdasarkan pertimbangan jumlah penduduk, jumlah pemilih terbaru, penyebarannya pemilih seperti tersebar dalam berapa kelurahan, dan sebagainya. Singkatnya, proporsional kalau pemilih banyak lokasi sampel (TPS) yang diambil pun banyak serta mewakili karakteristik populasi.
3. Mempersiapkan relawan untuk mengambil sampel dan meng-input-kannya ke sistem data. Jumlah relawan ini cukup banyak untuk mengambil data dari TPS yang telah dipilih.
   Pengambilan data sampel dilakukan oleh para relawan di beberapa TPS yang dipilih secara random.

Dalam memilih TPS yang akan dijadikan sampel pada *quick count* harus melewati beberapa tahap perhitungan. Diantanya adalah jumlah sampel yang harus dipenuhi, seperti jumlah pemilih dan jumlah TPS yang dibutuhkan. Besar sampel tergantung kepada 4 hal, yaitu keragaman (variasi) dari populasi, batas kesalahan sampel yang dikehendaki (*sampling eror*), interval kepercayaan (*confidence interval)*, dan besarnya populasi. Empat hal ini saling berkaitan dan berhubungan dan menjadi bagian dari rumus menentukan besar sampel.

Rumus yang digunakan dalam menentukan jumlah sampel adalah sebagai berikut.

$$n = \frac{Z^2.[p(1-p)].N}{Z^2.[p(1-p)] + (N-1).E^2}$$

Z = Tingkat kepercayaan yang dipakai
P(1-p)= Variasi Populasi
E = Kesalahan sampel yang dikehendaki
N = Jumlah populasi

Setelah jumlah pemilih didapatkan, maka jumlah TPS juga dapat ditentukan melalui rumus berikut ini.

$$SampelTPS = \frac{\text{jumlah pemilih yang dibutuhkan}}{\text{rata pemilih pada TPS}}$$

Proses selanjutnya adalah menentukan TPS yang akan dijadikan sampel. Metode yang biasanya digunakan oleh lembaga terkait adalah dengan mengunakan dasar stratifikasi. Sampel stratifikasi (stratified random sampling) merupakan teknik penarikan sampel dengan sampling unit dikelompokkan menjadi beberapa strata (kelompok) sehingga sampling unit dalam satu strata relatif homogen (Scheaffer et al. 1990). Secara sederhana dapat digambarkan melalui stratifikasi dengan dasar kecamatan. Dengan dasar stratifikasi kecamatan maka jumlah sample per kecamatan akan proporsional terhadap jumlah pemilihnya. Semakin banyak pemilih di setiap kecamatan maka samplenya juga semakin besar. Selanjutnya jumlah pemilih sampel kita konversi ke jumlah TPS.

Selanjutnya untuk menentukan TPS terpilih, dapat diterapkan teknik sampling probability, baik dengan simple random sampling atau systematic random sampling. Systematic random sampling adalah metode penentuan TPS berdasarkan interval sampel, dilakukan dengan membagi jumlah populasi TPS dengan jumlah TPS sampel (misal angka interval sampel yang didapat 10). Sampel pertama dipilih sacara acak dari TPS 1 hingga 10 (interval yang didapat). Misalnya dipilih angka 5. Selanjutnya, untuk memilih sampel kedua dan selanjutnya, bisa dilakukan dengan menambahkan 10 setelah sampel awal. Misal, sampel 1= TPS 5, sampel 2=TPS(10+5), sampel 3=TPS(15+10) begitu seterusnya hingga didapat 10 sampel.

Setelah semua TPS berhasil ditentukan maka relawan disebar untuk menghitung perolehan suara di TPS sampel dan kemudian mengirim data ke server melalui sms dengan format tertentu. Data yang telah didapat akan diolah di pusat data dengan menerapan ilmu stasistik, dari olahan data inilah lembaga survei bisa menghitung secara cepat siapa pemenang pemilu.

B. ALGORITMA *GREEDY*

Algoritma *greedy* adalah salah satu yang paling banyak digunakan dalam masalah optimasi. *Greedy* berarti „tamak‟ atau „rakus‟. Penamaan tersebut dikarenakan prinsip utama algoritma ini adalah mengambil solusi yang paling baik pada saat itu juga. Solusi yang baik ini adalah solusi yang memberikan sumbangan paling berarti (memiliki bobot paling kecil ataupun keuntungan yang paling besar) dan masih

memenuhi syarat-syarat yang diberlakukan dalam pengambilan.

Dalam menyelesaikan persoalan, algoritma *greedy* akan terdiri dari beberapa tahap. Untuk setiap tahap, terdapat sebuah solusi yang juga disebut sebagai optimum lokal.(karena dinyatakan optimum hanya untuk kondisi saat itu juga). Nilai-nilai optimum lokal inilah yang, setelah digabungkan, diharapkan akan menjadi optimum global (alias nilai optimum yang diicapai untuk persoalan tersebut).

Untuk menggunakan algoritma *greedy* dalam menyelesaikan suatu persoalan, terlebih dahulu haruslah didefinisikan elemen-element algoritma *greedy* yang berlaku dalam persoalan tersebut. Adapun elemen-elemen yang dimiliki algoritma *greedy* antara lain:

- Himpunan kandidat (C)
- Himpunan solusi (S)
- Fungsi seleksi
- Fungsi kelayakan
- Fungsi obyektif

Pembahasan mengenai elemen-elemen tersebut secara lebih mendetail antara lain:

1. Himpunan Kandidat

Himpunan kandidat merupakan himpunan yang mengandung elemen-elemen yang dapat menjadi bagian dari solusi. Contoh untuk himpunan kandidat adalah koin-koin dalam persoalan optimasi penukaran uang dengan sejumlah koin.

2. Himpunan Solusi

Himpunan solusi terdiri dari elemen-elemen yang sesungguhnya merupakan bagian dari himpunan kandidat dan apabila dikumpulkan bersama dapat menjadi solusi global dari permasalahan yang ada. Contoh himpunan solusi adalah kombinasi uang logam berjumlah paling sedikit namun memiliki nilai yang diinginkan dalam persoalan optimasi penukaran uang.

3. Fungsi Seleksi

Fungsi inilah yang berperan dalam pemilihan solusi lokal setiap tahap dalam algoritma *greedy*. Solusi lokal pada tahap tersebut adalah nilai yang dihasilkan oleh fungsi seleksi setelah digunakan pada himpunan kandidat. Contoh fungsi seleksi ini adalah memilih koin dengan nilai paling tinggi di antara himpunan kandidat yang disediakan.

4. Fungsi Kelayakan

Fungsi yang digunakan untuk mengetahui apakah solusi yang optimal itu masih sesuai dengan syarat yang diminta dalam persoalan. Fungsi kelayakan merupakan pembatas dalam pemanfaatan fungsi seleksi, karena hasil fungsi seleksi haruslah dinyatakan layak

5. Fungsi Obyektif

Optimasi yang diharapkan dari pemecahan masalah dengan menggunakan algoritma *greedy*. Biasanya bersifat mencari yang sekecil-kecilnya atau mencari yang sebesar-bersarnya. Untuk masalah penukaran uang, contoh fungsi obyektifnya adalah jumlah koin yang digunakan minimum.

Berikut ini merupakan psedocode dari algoritma *greedy*.

```
  procedure greedy(input C: himpunan_kandidat;
  output S : himpunan_solusi)
  { menentukan solusi optimum dari persoalan optimasi
  dengan algoritma greedy. Masukan: himpunan kandidat C.
  Keluaran: himpunan solusi S
  }
  Deklarasi
   x : kandidat;

  Algoritma:
   S←{}
   { inisialisasi S dengan kosong }
   while (belum SOLUSI(S)) and (C ≠ {} ) do
   x←SELEKSI(C);
   { pilih sebuah kandidat dari C}
   C← C - {x}
   { elemen himpunan kandidat berkurang satu }
   if LAYAK(S ∪ {x}) then
   S←S ∪ {x}
   endif
   endwhile
```

III. IMPLEMENTASI *GREEDY* PADA PENENTUAN TPS

Penentuan TPS untuk *Quick count* biasanya dilakukan dengan metode random dengan berdasarkan interval. Dalam tulisan ini, penulis akan mencoba menerapkan algoritma *greedy* dalam menentukan TPS yang akan dijadikan sebagai sampel.

Elemen algoritma *greedy* untuk permasalahan ini adalah sebagai berikut :

- Himpunan kandidat (C) : semua TPS yang ada pada daerah pemilihan
- Himpunan solusi (S) : TPS yang dipilih sebagai sampel
- Fungsi seleksi : memilih TPS dengan jumlah pemilih yang memiliki selisih terkecil dengan jumlah rata-rata pemilih pada TPS
- Fungsi kelayakan : jumlah TPS belum melebihi jumlah kebutuhan dan TPS belum pernah dipilih sebelumnya
- Fungsi Objektif : mendapatkan daftar TPS dengan total jumlah pemilih mendekati jumlah sampel pemilih pada daerah tersebut

Contoh penerapannya dapat dilihat pada kasus berikut ini. Contoh kasus mengacu kepada http://politik.kompasiana.com/2014/04/13/quick-count-pileg-2014-konspirasi-media-bag2-646584.html dengan beberapa modifikasi. Misalnya untuk mengadakan *quick count* di kota Padang dengan jumlah pemilih 560.732 orang dan jumlah TPS sebanyak 1.532 TPS. Sesuai dengan metodologi maka sample pemilih sebanyak 89.252

orang atau 244 TPS. Dasar stratifikasinya adalah proporsi kecamatan dengan 11 kecamatan di Kota Padang.

Berikut ilustrasinya :

Total pemilih		560,732			Jumlah TPS	1,532
Sampel pemilih		89,252			Total Pemilih/Jml TPS	336
no	Kecamatan	Jumlah TPS	Jumlah Pemilih	Proporsi	Sampel Pemilih	Sampel TPS
1	Padang Selatan	122	33,574	6%	5,177	18
2	Padang Timur	145	55,122	10%	8,767.30	24
3	Padang Barat	95	32,318	6%	5,140.30	14
4	Padang Utara	113	37,314	7%	5,934.90	16
5	Bungus Teluk Kabung	46	15,936	3%	2,534.70	7
6	Lubuk Begalung	183	72,091	13%	11,466.30	31
7	Lubuk Kilangan	82	31,948	6%	5,081	14
8	Pauh	105	37,544	7%	5,972	16
9	Kuranji	248	87,421	16%	13,904.60	38
10	Nanggalo	101	37,087	7%	5,898.80	16
11	Koto Tangah	292	113,091	20%	17,987.50	49

Gambar 2. Data TPS pada Kota Padang

TPS yang ada pada kecamatan Padang Selatan diataranya adalah sebagai berikut.

No TPS	Total Pemilih
1	241
2	334
3	239
4	279
5	314
6	294
7	268
8	293
9	263
10	239
11	399
12	256
13	295
14	255
15	214
16	200
17	297
18	241
19	334
20	239

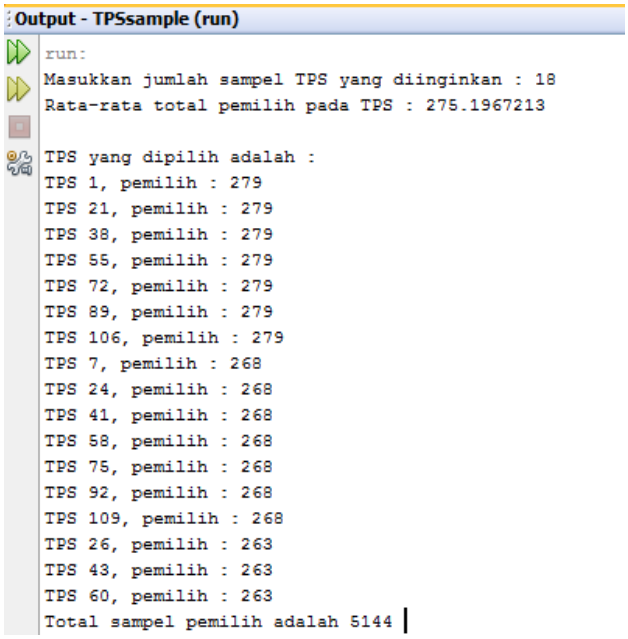
Gambar 3. Data jumlah pemilih di setiap TPS

Rata-rata total pemilih setiap TPS untuk kecamatan Padang Selatan adalah 275,19 orang. Maka TPS sampel akan ditentukan melalui algoritma *greedy* by value (mengacu pada jumlah pemilih TPS yang mendekati rata-rata). TPS yang dijadikan sampel berjumlah 18 TPS.

Untuk menerapkan algoritma *greedy* pada penentuan TPS, pengujiannya akan dibatu oleh program sederhana. Program ini akan membaca file txt yang berisi daftar TPS beserta jumlah pemilihnya, kemudian akan dimasukkan ke dalam array. Dari TPS yang diinputkan akan didapatkan rata-rata jumlah pemilih pada semua TPS. Kemudian dibuat array baru yang menampung selisih antara jumlah pemilih pada TPS dengan rata-rata jumlah pemilih.

Algoritma *greedy* akan mengambil TPS dengan selisih terkecil kemudian memeriksanya apakah TPS tersebut layak dan jumlah TPS masih kurang dari 18.

Hasil Implementasi pada program adalah sebagai berikut.



Gambar 4. Hasil Implementasi Program

Dari hasil pengujian algoritma *greedy* didapatkan sample pemilih sebanyak 5144 orang dari 18 TPS. Angka ini memiliki sedikit perbedaan dengan sampel pemilih yang dibutuhkan yaitu 5177,61 orang.

IV. Analisis Hasil Pengujian

Hasil pengujian yang didapatkan adalah 18 TPS dengan total pemilih sebesar 5144. Hasil ini masih memiliki perbedaan dengan target kebutuhan. Hal tersebut mungkin disebabkan karena algoritma *greedy* hanya memperhitungkan solusi-solusi yang elemennya merupakan solusi terbaik dalam tahapan tertentu, optimum global belum tentu menjadi solusi paling baik yang mungkin didapatkan. Masih ada kemungkinan bahwa solusi yang tidak diperhitungkan dengan menggunakan algoritma *greedy* justru merupakan solusi yang paling optimum untuk permasalahan yang ada. Oleh karena itu, algoritma *greedy* lebih banyak digunakan untuk mencari hampiran terhadap sebuah solusi. Untuk mendapatkan solusi yang paling mendekati solusi sebenarnya, pendefinisian fungsi seleksi harus dilakukan dengan hati-hati. Penggunaan algorima lain seperti proram dinamis mungkin dapat menghasilkan solusi lebih baik.

Namun dibandingkan dengan pemilihan secara random seperti yang biasa dilakukan, algoritma *greedy* dapat menghasilkan solusi yang lebih baik.

V. KESIMPULAN

Algoritma *greedy* merupakan salah satu strategi untuk menyelesaikan persoalan optimasi. Hasil dari algoritma ini dapat mendekati hasil yang benar-benar paling optimal, walaupun terkadang solusi yang diberikannya hanya mendekati optimal. Algoritma *greedy* dapat digunakan dalam metode *quick count* pada tahap penentuan sampel TPS.

REFERENCES

[1] Rinaldi Munir, "Algoritma *Greedy*" Diktat Kuliah IF3051 Strategi Algoritma, Bandung: Program Studi Teknik Informatika, 2009, hal.41-84.

[2] http://www.cyrusnetwork.co/cyrus/home/services/e4da3b7fbbce2345d7772b0674a318d5/quick-count#sthash.MBG2jSck.dpbs

[3] http://politik.kompasiana.com/2014/04/10/quick-count-pileg-2014-konspirasi-media--646153.html

[4] http://politik.kompasiana.com/2014/04/13/quick-count-pileg-2014-konspirasi-media-bag2-646584.html

[5] http://staff.uny.ac.id/sites/default/files/penelitian/Kismiantini,%20/A2007_B1.pdf

[6] http://www.suarapembaruan.com/politikdanhukum/quick-count-itu-bagian-dari-alat-kontrol/52604.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Mei 2014

Dwitri Desvira
13510044