Penerapan Algoritma Branch and Bound untuk Optimasi Jadwal Pertemuan Berkelompok agar Setiap Pasangan Peserta Bertemu Minimal Sekali

Muhammad Faiz Alfikrona - 18223009
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: faizalfikrona@gmail.com , 18223009@std.stei.itb.ac.id

***Abstrak*—Permasalahan penjadwalan pertemuan berkelompok muncul ketika sejumlah peserta harus saling bertemu melalui meeting berkapasitas terbatas. Setiap slot waktu dapat memuat beberapa meeting secara paralel, tetapi seorang peserta hanya boleh berada pada satu meeting dalam slot yang sama. Makalah ini memodelkan persoalan tersebut sebagai optimasi kombinatorial dengan tujuan utama meminimalkan jumlah slot waktu sehingga setiap pasangan peserta bertemu minimal sekali. Tujuan sekunder adalah meminimalkan pengulangan pasangan, karena jadwal ideal membuat setiap pasangan bertemu tepat sekali. Algoritma Branch and Bound diterapkan dengan state berupa jadwal parsial, himpunan pasangan yang telah tercakup, dan penalti pengulangan. Fungsi bound dibangun dari jumlah pasangan tersisa serta jumlah peserta yang belum ditemui oleh setiap peserta. Hasil pengujian pada beberapa ukuran kecil menunjukkan bahwa bound mampu membuktikan optimalitas jumlah slot waktu dan memangkas sebagian cabang pencarian melalui lower bound dan memoization.**

Kata kunci—Branch and Bound; optimasi jadwal; meeting paralel; cakupan pasangan; Social Golfer Problem.

I. PENDAHULUAN

Kegiatan diskusi kelas, mentoring, networking, orientasi, dan lokakarya sering menggunakan mekanisme pertemuan berkelompok. Dalam mekanisme tersebut, peserta tidak selalu dapat bertemu secara bebas; interaksi hanya terjadi ketika peserta berada dalam meeting yang sama. Jika jumlah peserta cukup banyak, penyusunan jadwal secara manual menjadi sulit karena penyusun harus memastikan bahwa semua pasangan peserta memperoleh kesempatan bertemu minimal satu kali.

Masalah yang dibahas pada makalah ini adalah sebagai berikut. Terdapat a orang peserta dan setiap meeting hanya dapat memuat paling banyak b orang. Pada satu slot waktu dapat dilaksanakan nol atau banyak meeting secara paralel. Akan tetapi, seorang peserta hanya dapat mengikuti paling banyak satu meeting dalam satu slot waktu. Tujuan utama penjadwalan adalah meminimalkan jumlah slot waktu yang diperlukan sampai seluruh pasangan peserta pernah berada dalam meeting yang sama.

Fokus pada jumlah slot waktu penting karena beberapa meeting dapat berlangsung paralel. Sebagai contoh, untuk empat peserta A, B, C, dan D dengan kapasitas dua orang, semua pasangan memang membutuhkan enam meeting: AB, AC, AD, BC, BD, dan CD. Namun keenam meeting tersebut dapat disusun dalam tiga slot waktu: slot pertama AB dan CD, slot kedua AC dan BD, serta slot ketiga AD dan BC. Dengan demikian, jumlah meeting tidak secara langsung menggambarkan lama jadwal.

Selain memenuhi cakupan minimal satu kali, jadwal yang baik juga sebaiknya menghindari pengulangan pasangan yang sama. Kondisi ideal adalah setiap pasangan bertemu tepat satu kali. Namun kondisi ini tidak selalu mungkin untuk semua kombinasi jumlah peserta dan kapasitas meeting. Oleh karena itu, pengulangan pasangan diperlakukan sebagai penalti yang diminimalkan setelah tujuan utama jumlah slot waktu terpenuhi.

Persoalan ini bersifat kombinatorial. Pada setiap slot waktu terdapat banyak kemungkinan konfigurasi meeting paralel, dan jumlah kemungkinan meningkat tajam ketika a bertambah. Algoritma Branch and Bound dipilih karena dapat melakukan pencarian sistematis, tetapi tetap memangkas cabang yang secara matematis tidak mungkin menghasilkan solusi yang lebih baik daripada solusi terbaik sementara.

Kontribusi makalah ini adalah pemodelan formal persoalan penjadwalan meeting berkapasitas terbatas sebagai persoalan cakupan pasangan, rancangan algoritma Branch and Bound berbasis konfigurasi slot waktu, serta pengujian awal untuk membandingkan solusi Branch and Bound dengan pendekatan greedy sederhana.

II. DASAR TEORI

A. *Optimasi Kombinatorial*

Optimasi kombinatorial adalah persoalan mencari solusi terbaik dari ruang solusi diskrit yang biasanya sangat besar. Berbeda dengan optimasi kontinu, keputusan pada optimasi kombinatorial sering berupa pemilihan, pengelompokan, urutan, atau penjadwalan [6], [7]. Pada persoalan ini, keputusan yang harus dipilih adalah konfigurasi meeting pada setiap slot

waktu. Setiap konfigurasi menghasilkan sejumlah pasangan peserta yang tercakup.

Jika semua kemungkinan konfigurasi diperiksa tanpa pemangkasan, pendekatan tersebut setara dengan brute force. Brute force berguna untuk kasus sangat kecil karena dapat menjadi pembanding kebenaran, tetapi tidak praktis untuk ukuran yang lebih besar. Hal ini terjadi karena jumlah cara membentuk kelompok dari a peserta dengan kapasitas b meningkat secara kombinatorial [2].

B. *Branch and Bound*

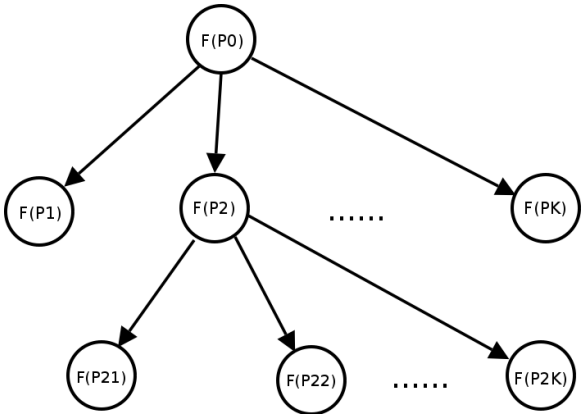

Fig. 1. Ilustrasi pemecahan masalah menjadi subpersoalan pada Branch and Bound. Sumber: Wikimedia Commons, Enrico Strocchi, CC BY-SA 3.0/GFDL.

Branch and Bound adalah strategi algoritma untuk menyelesaikan persoalan optimasi diskrit [1]. [2]. Branching dilakukan dengan membagi persoalan menjadi subpersoalan, sedangkan bounding digunakan untuk menghitung batas optimistis dari kualitas solusi yang masih mungkin diperoleh pada suatu cabang. Jika batas tersebut tidak dapat mengalahkan solusi terbaik sementara, cabang tersebut dipangkas.

Pada makalah ini, sebuah simpul pencarian merepresentasikan jadwal parsial. Cabang baru dibentuk dengan menambahkan satu konfigurasi slot waktu yang valid. Nilai bound memperkirakan minimal jumlah slot waktu tambahan yang masih diperlukan untuk menutup pasangan peserta yang belum pernah bertemu.

C. *Hubungan dengan Social Golfer Problem*

Masalah ini berhubungan dengan Social Golfer Problem, yaitu persoalan menyusun beberapa putaran permainan sehingga pemain ditempatkan pada kelompok dan tidak ada pasangan yang berada dalam kelompok yang sama lebih dari satu kali [4], [5]. Perbedaannya, Social Golfer Problem klasik biasanya berfokus pada memaksimalkan jumlah putaran tanpa pengulangan pasangan. Persoalan pada makalah ini berfokus pada menutup semua pasangan minimal satu kali dan meminimalkan jumlah slot waktu. Dengan demikian, pengulangan pasangan diperbolehkan jika diperlukan, tetapi dihitung sebagai penalti.

D. *Hubungan dengan Set Cover*

Persoalan juga dapat dipandang sebagai variasi Set Cover. Himpunan semesta adalah semua pasangan peserta yang harus tercakup. Setiap konfigurasi slot waktu merepresentasikan sebuah subset dari pasangan yang dapat bertemu pada slot tersebut. Tujuannya adalah memilih sejumlah konfigurasi slot waktu sesedikit mungkin sehingga gabungan subset tersebut mencakup seluruh pasangan [3], [6]. Perbedaannya, subset tidak bebas dibentuk karena harus memenuhi kendala kapasitas meeting dan kendala bahwa peserta tidak boleh muncul di dua meeting pada slot yang sama.

III. Pemodelan Persoalan

A. *Definisi Peserta, Meeting, dan Slot Waktu*

Misalkan P = {1, 2, ..., a} adalah himpunan peserta. Kapasitas maksimum sebuah meeting adalah b orang. Sebuah meeting M adalah subset dari P dengan ukuran 2 <= |M| <= b. Meeting berukuran satu orang tidak dipertimbangkan karena tidak menghasilkan pasangan baru. Satu slot waktu T_t adalah kumpulan meeting yang saling lepas, sehingga setiap peserta muncul paling banyak sekali pada slot tersebut.

Himpunan semua pasangan yang wajib tercakup dinyatakan sebagai R = {(i, j) | i, j dalam P dan i < j}. Jumlah pasangan wajib adalah C(a, 2) = a(a - 1)/2. Jika peserta i dan j berada pada meeting yang sama, maka pasangan (i, j) dianggap tercakup. Jadwal dikatakan feasible apabila setiap pasangan dalam R tercakup minimal satu kali.

TABLE I. Notasi Pemodelan Persoalan

Notasi	Makna
a	Jumlah peserta
b	Kapasitas maksimum setiap meeting
P	Himpunan peserta
R	Himpunan semua pasangan peserta
M	Satu meeting, berisi 2 sampai b peserta
T_t	Kumpulan meeting paralel pada slot waktu ke-t
cover(M)	Pasangan yang tercakup oleh meeting M
cover(T_t)	Gabungan pasangan yang tercakup pada slot waktu T_t

B. *Kendala*

Kendala pertama adalah kapasitas meeting. Untuk setiap meeting M, berlaku |M| <= b. Kendala kedua adalah konflik waktu. Untuk dua meeting berbeda M_x dan M_y pada slot waktu yang sama, berlaku M_x dan M_y beririsan kosong. Kendala ketiga adalah cakupan pasangan. Untuk setiap pasangan (i, j) dalam R, harus terdapat minimal satu slot waktu dan satu meeting di slot tersebut yang memuat i dan j sekaligus.

Slot waktu kosong diperbolehkan menurut definisi masalah, tetapi tidak pernah dibangkitkan oleh algoritma karena slot kosong tidak menambah cakupan pasangan. Jika tujuan adalah meminimalkan jumlah slot waktu, menambahkan slot kosong hanya membuat solusi menjadi lebih buruk.

C. *Fungsi Objektif*

Fungsi objektif dirancang secara leksikografis. Prioritas pertama adalah meminimalkan jumlah slot waktu. Prioritas kedua adalah meminimalkan penalti pengulangan pasangan. Penalti suatu pasangan bernilai nol jika pasangan tersebut bertemu tepat satu kali, dan bernilai f(i,j) - 1 jika pasangan tersebut bertemu sebanyak f(i,j) kali. Dengan cara ini, jadwal dengan slot waktu lebih sedikit selalu lebih diprioritaskan daripada jadwal dengan repetisi lebih rendah tetapi slot waktu lebih banyak.

Secara ringkas, optimasi dilakukan dalam urutan: minimalkan jumlah slot waktu, penuhi semua pasangan minimal satu kali, lalu minimalkan jumlah pengulangan pasangan. Jumlah meeting total dapat dicatat sebagai metrik tambahan, tetapi bukan target utama karena meeting dapat dijalankan secara paralel.

D. *Lower Bound Teoretis*

Dua lower bound digunakan untuk memperkirakan minimal slot tambahan. Bound pertama berasal dari jumlah pasangan tersisa. Jika terdapat r pasangan yang belum tercakup, sedangkan satu slot waktu paling banyak dapat mencakup q pasangan baru, maka minimal slot tambahan adalah ceil(r / q). Nilai q diperoleh dari susunan meeting yang memaksimalkan jumlah pasangan dalam satu slot waktu dengan kapasitas b.

Bound kedua berasal dari sudut pandang tiap peserta. Dalam satu slot waktu, seorang peserta yang masuk meeting berukuran paling banyak b hanya dapat bertemu dengan paling banyak b - 1 peserta baru. Jika peserta x masih belum bertemu dengan u_x peserta lain, maka peserta tersebut membutuhkan minimal ceil(u_x / (b - 1)) slot tambahan. Lower bound global adalah nilai maksimum dari bound pasangan dan seluruh bound peserta.

IV. RANCANGAN ALGORITMA

A. *Representasi State*

Sebuah state pada pohon pencarian terdiri atas jadwal parsial, himpunan pasangan yang telah tercakup, jumlah slot waktu yang telah digunakan, dan nilai penalti pengulangan. Informasi penting yang digunakan untuk pruning adalah himpunan pasangan tercakup. Dua jadwal parsial yang berbeda tetapi menghasilkan himpunan pasangan tercakup yang sama memiliki potensi lanjutan yang serupa, sehingga memoization dapat digunakan untuk menghindari eksplorasi ulang state yang lebih buruk.

B. *Pembangkitan Konfigurasi Slot Waktu*

Branching dilakukan per slot waktu, bukan per meeting. Pada setiap langkah, algoritma membangkitkan semua konfigurasi slot waktu yang valid. Sebuah konfigurasi terdiri atas satu atau lebih meeting yang saling lepas, dan setiap meeting memiliki ukuran antara dua sampai b. Pendekatan ini membuat kendala konflik waktu otomatis terpenuhi pada setiap cabang.

Konfigurasi slot waktu kemudian diurutkan berdasarkan banyaknya pasangan baru yang ditambahkan. Konfigurasi yang menutup lebih banyak pasangan baru dieksplorasi lebih dahulu agar solusi feasible cepat ditemukan. Solusi feasible awal juga dapat diperoleh dengan greedy, yaitu berulang kali memilih konfigurasi slot yang memberikan tambahan pasangan baru terbanyak.

C. *Fungsi Bound dan Pruning*

Misalkan best adalah jumlah slot waktu terbaik yang telah ditemukan. Pada suatu state, algoritma menghitung lower bound slot tambahan. Jika slot_used + lower_bound lebih besar daripada best, maka cabang dipangkas. Jika nilainya sama dengan best tetapi penalti pengulangan sementara sudah tidak mungkin lebih baik, cabang juga dapat dipangkas untuk optimasi sekunder.

Pruning ini aman untuk objektif utama karena lower bound tidak pernah melebih-lebihkan jumlah slot tambahan minimal. Dengan kata lain, jika lower bound menyatakan bahwa suatu cabang tetap membutuhkan terlalu banyak slot, maka tidak ada solusi turunan dari cabang tersebut yang dapat mengalahkan solusi terbaik sementara.

D. *Pseudocode*

Variabel COST menyatakan jumlah slot waktu yang telah digunakan, FREQ[p] menyimpan banyaknya pertemuan untuk pasangan p, sedangkan MEMO menyimpan state terbaik yang sudah pernah dikunjungi. Algoritma mengoptimalkan solusi secara leksikografis: pertama jumlah slot waktu, kemudian jumlah pengulangan pasangan.

```
BRANCH-AND-BOUND-SCHEDULE(P, b)
1  R <- GENERATE-ALL-PAIRS(P)
2  CANDIDATES <- GENERATE-ALL-VALID-SLOTS(P, b)
3  BEST-SCHEDULE <- GREEDY-SCHEDULE(P, b, CANDIDATES)
4  BEST-COST <- LENGTH(BEST-SCHEDULE)
5  BEST-REPEAT <- REPEATED-PAIRS(BEST-SCHEDULE)
6  MEMO <- empty map
7  FREQ[p] <- 0 for each p in R
8  SEARCH(<>, FREQ, 0, R, CANDIDATES)
9  return BEST-SCHEDULE

SEARCH(S, FREQ, COST, R, CANDIDATES)
1  COVERED <- {p in R : FREQ[p] > 0}
2  REP_NOW <- TOTAL-REPETITION(FREQ)
3  if COVERED in MEMO and MEMO[COVERED] <= (COST, REP_NOW)
4      return
5  MEMO[COVERED] <- (COST, REP_NOW)
6  if COVERED = R
7      if COST < BEST-COST or
8         (COST = BEST-COST and REP_NOW < BEST-REPEAT)
9          BEST-COST <- COST
10         BEST-REPEAT <- REP_NOW
11         BEST-SCHEDULE <- S
12     return
13 REMAINING <- R - COVERED
14 LB <- LOWER-BOUND(REMAINING, P, b)
15 if COST + LB > BEST-COST
16     return
17 SORT-CANDIDATES-BY-NEW-COVERAGE(CANDIDATES, FREQ)
18 for each slot T in CANDIDATES
19     NEW <- PAIRS-COVERED-BY-SLOT(T)
20     GAIN <- {p in NEW : FREQ[p] = 0}
21     if GAIN = empty set
22         continue
23     FREQ' <- COPY(FREQ)
24     for each pair p in NEW
25         FREQ'[p] <- FREQ'[p] + 1
26     S' <- APPEND(S, T)
27     SEARCH(S', FREQ', COST + 1, R, CANDIDATES)

LOWER-BOUND(REMAINING, P, b)
1  a <- |P|
2  r <- |REMAINING|
3  q <- MAX-PAIRS-PER-SLOT(a, b)
4  LB1 <- CEIL(r / q)
5  LB2 <- 0
6  for each participant x in P
7      dx <- number of pairs in REMAINING that contain x
8      LB2 <- MAX(LB2, CEIL(dx / (b - 1)))
9  return MAX(LB1, LB2)

MAX-PAIRS-PER-SLOT(a, b)
1  m <- FLOOR(a / b)
2  s <- a mod b
3  q <- m * C(b, 2)
4  if s >= 2
5      q <- q + C(s, 2)
6  return q
```

```
GENERATE-ALL-VALID-SLOTS(P, b)
1  CANDIDATES <- empty set
2  generate every collection T = {M1, M2, ..., Mk}
3       such that 2 <= |Mi| <= b for each meeting Mi
4       and Mi intersect Mj = empty set for i != j
5  insert every valid T into CANDIDATES
6  return CANDIDATES
```

E. *Contoh Kecil*

Untuk a = 6 dan b = 3, terdapat 15 pasangan yang harus tercakup. Salah satu solusi dengan empat slot waktu dan tanpa pengulangan pasangan ditunjukkan pada Table II. Slot pertama memakai dua meeting berukuran tiga. Tiga slot berikutnya memakai meeting berukuran dua untuk menutup pasangan lintas kelompok. Solusi ini menunjukkan bahwa meeting tidak harus selalu berukuran maksimum; ukuran meeting yang lebih kecil dapat membantu menghindari pengulangan pasangan.

TABLE II. CONTOH JADWAL UNTUK A=6 DAN B=3

Slot	Meeting paralel
1	{A,B,C},{D,E,F}
2	{A,D},{B,E},{C,F}
3	{A,E},{B,F},{C,D}
4	{A,F},{B,D},{C,E}

F. *Ilustrasi Perhitungan Bound*

Pada contoh a = 6 dan b = 3, jumlah pasangan total adalah C(6, 2) = 15. Dalam satu slot waktu, konfigurasi dengan dua meeting berukuran tiga dapat mencakup paling banyak 2 x C(3, 2) = 6 pasangan. Oleh karena itu, lower bound berbasis pasangan pada awal pencarian adalah ceil(15 / 6) = 3 slot. Bound ini optimistis karena belum memperhitungkan susunan pasangan yang saling berbagi peserta.

Setelah slot pertama {A,B,C}, {D,E,F}, terdapat enam pasangan yang tercakup, yaitu AB, AC, BC, DE, DF, dan EF. Sembilan pasangan lain merupakan pasangan lintas dua kelompok awal. Karena setiap peserta hanya dapat bertemu paling banyak dua orang baru dalam satu slot berkapasitas tiga, secara individual setiap peserta masih membutuhkan minimal ceil(3 / 2) = 2 slot tambahan. Maka lower bound setelah slot pertama juga bernilai dua slot tambahan.

Namun, memilih kembali meeting berukuran tiga pada slot kedua dapat membuat banyak pasangan berulang. Algoritma dapat memilih konfigurasi yang berisi meeting berukuran dua, misalnya {A,D}, {B,E}, dan {C,F}. Konfigurasi ini hanya menutup tiga pasangan baru, tetapi tidak membuat pengulangan. Keputusan semacam ini menunjukkan bahwa konfigurasi dengan cakupan terbesar secara lokal belum tentu terbaik untuk tujuan sekunder.

Dari ilustrasi tersebut, terlihat bahwa fungsi bound dipakai untuk membatasi jumlah slot, sedangkan pengurutan cabang dan penalti repetisi membantu memilih jadwal yang lebih rapi. Branch and Bound tetap menjamin pencarian sistematis karena seluruh konfigurasi valid masih dapat dipertimbangkan selama tidak dipangkas oleh bound yang sah.

V. ANALISIS KOMPLEKSITAS DAN KARAKTERISTIK MASALAH

A. *Ukuran Ruang Pencarian*

Jumlah kandidat meeting untuk a peserta dan kapasitas b adalah jumlah kombinasi C(a, s) untuk s dari dua sampai b. Nilai ini bertambah cepat ketika b meningkat. Setelah kandidat meeting diperoleh, algoritma masih harus membentuk konfigurasi slot waktu, yaitu kumpulan meeting yang saling lepas. Pembentukan konfigurasi slot waktu ini setara dengan memilih beberapa subset yang tidak beririsan, sehingga jumlah konfigurasinya dapat jauh lebih besar daripada jumlah meeting tunggal.

Sebagai contoh, ketika a = 8 dan b = 4, terdapat banyak meeting berukuran dua, tiga, dan empat. Dari meeting-meeting tersebut dapat dibentuk ribuan konfigurasi slot yang valid. Setiap konfigurasi slot menjadi pilihan cabang pada pohon pencarian. Tanpa bound dan memoization, urutan pemilihan slot yang berbeda dapat menghasilkan jadwal dengan cakupan pasangan yang sama, sehingga eksplorasi menjadi sangat redundan.

Ruang pencarian juga memiliki banyak simetri. Menukar label peserta atau menukar urutan meeting dalam satu slot sering menghasilkan jadwal yang secara struktur sama. Implementasi dasar pada makalah ini mengurangi sebagian simetri dengan membangkitkan konfigurasi meeting dalam urutan kanonis, tetapi belum melakukan symmetry breaking yang agresif. Karena itu, masih terdapat peluang optimasi yang besar untuk versi lanjut.

B. *Kompleksitas Waktu*

Pada kasus terburuk, Branch and Bound tetap memiliki kompleksitas eksponensial karena algoritma dapat harus mencoba kombinasi konfigurasi slot yang sangat banyak. Hal ini tidak dapat dihindari untuk pencarian solusi optimal pada persoalan kombinatorial umum. Akan tetapi, performa praktis dapat jauh lebih baik apabila lower bound kuat dan solusi awal yang menjadi upper bound cukup dekat dengan optimum.

Faktor yang paling memengaruhi waktu eksekusi adalah jumlah kandidat slot dan kedalaman solusi optimal. Kedalaman solusi optimal sama dengan jumlah slot minimum yang dicari. Jika kapasitas b kecil, setiap slot menutup sedikit pasangan sehingga kedalaman bertambah. Jika b besar, kedalaman dapat turun, tetapi jumlah kandidat meeting dan potensi repetisi meningkat. Oleh karena itu, perubahan b tidak selalu membuat persoalan menjadi lebih mudah.

Penggunaan bitmask membantu menurunkan biaya operasi himpunan. Operasi union, intersection, dan difference terhadap pasangan dapat dilakukan dengan operasi bit. Walaupun demikian, bitmask hanya mempercepat evaluasi sebuah state; ia tidak mengubah fakta bahwa jumlah state potensial tetap besar.

C. *Kompleksitas Memori*

Memori terutama digunakan untuk menyimpan daftar kandidat slot, tabel memoization, dan jadwal terbaik sementara. Daftar kandidat slot dapat menjadi besar karena setiap konfigurasi slot menyimpan daftar meeting dan bitmask pasangan yang tercakup. Memoization menyimpan state berdasarkan himpunan pasangan tercakup agar state yang sama tidak dieksplorasi ulang dengan jumlah slot atau penalti yang lebih buruk.

Untuk kasus kecil, pendekatan ini memadai. Untuk kasus besar, daftar kandidat slot sebaiknya tidak selalu dibangkitkan seluruhnya di awal. Alternatifnya adalah membangkitkan kandidat secara lazy sesuai kebutuhan, atau hanya membangkitkan sejumlah kandidat terbaik berdasarkan heuristik cakupan pasangan baru.

D. Keterbatasan Model

Model pada makalah ini belum memasukkan ketersediaan waktu peserta, preferensi pasangan, lokasi meeting, durasi meeting yang berbeda, atau batas jumlah meeting fisik yang dapat berjalan paralel. Dalam aplikasi nyata, faktor-faktor tersebut dapat mengubah persoalan menjadi lebih kompleks. Misalnya, jika hanya tersedia dua ruang meeting pada setiap slot, maka jumlah meeting paralel tidak hanya dibatasi oleh peserta, tetapi juga oleh sumber daya ruangan.

Model juga menganggap semua pasangan memiliki prioritas sama. Dalam beberapa konteks, pasangan tertentu mungkin lebih penting untuk dipertemukan lebih awal. Contoh lain adalah kegiatan mentoring, di mana peserta dari kategori tertentu harus dicampur secara merata. Kendala seperti ini dapat ditambahkan dengan memperluas fungsi objektif atau menambahkan constraint baru.

Keterbatasan lain adalah optimasi repetisi pasangan belum selalu mudah dibuktikan optimal untuk ukuran lebih besar. Dalam makalah ini, prioritas utama adalah pembuktian jumlah slot minimum. Optimasi repetisi dipakai sebagai tujuan sekunder dan bergantung pada seberapa jauh pohon pencarian dieksplorasi setelah jumlah slot minimum ditemukan.

E. Kasus Khusus

Beberapa nilai parameter menghasilkan bentuk persoalan yang lebih mudah dianalisis. Ketika b = 2, setiap meeting hanya berisi dua peserta. Persoalan berubah menjadi penjadwalan pasangan seperti round-robin tournament. Dalam satu slot waktu, pasangan yang dapat dijadwalkan harus membentuk matching, yaitu kumpulan sisi yang tidak berbagi simpul. Untuk a genap, satu slot dapat memuat a/2 meeting; untuk a ganjil, satu peserta harus menganggur pada setiap slot. Jumlah slot minimum pada kasus ini adalah a - 1 untuk a genap dan a untuk a ganjil.

Ketika b >= a, seluruh peserta dapat ditempatkan dalam satu meeting pada satu slot waktu. Semua pasangan langsung tercakup, sehingga solusi optimal adalah satu slot dengan satu meeting. Kasus ini menjadi batas bawah trivial dan dapat ditangani sebelum algoritma Branch and Bound dijalankan.

Kasus yang paling menarik berada di antara dua ekstrem tersebut, yaitu 2 < b < a. Pada rentang ini, satu meeting dapat menutup banyak pasangan, tetapi peserta yang berada dalam satu meeting tidak dapat mengikuti meeting lain pada slot yang sama. Akibatnya, terdapat trade-off antara membentuk meeting besar untuk menutup banyak pasangan dan membentuk meeting kecil untuk menghindari pengulangan pasangan.

F. Strategi Heuristik dalam Branching

Walaupun Branch and Bound bersifat eksak terhadap objektif utama, urutan eksplorasi cabang tetap berpengaruh besar terhadap waktu pencarian. Jika solusi baik ditemukan lebih awal, upper bound menjadi kecil dan cabang lain lebih mudah dipangkas. Oleh karena itu, kandidat slot diurutkan berdasarkan jumlah pasangan baru yang dapat ditutup. Jika terdapat dua kandidat dengan jumlah pasangan baru sama, kandidat dengan repetisi lebih sedikit diprioritaskan.

Heuristik lain yang dapat digunakan adalah memprioritaskan pasangan yang paling sulit ditutup. Misalnya, jika ada peserta yang masih belum bertemu dengan banyak peserta lain, konfigurasi yang melibatkan peserta tersebut dapat diberi prioritas lebih tinggi. Ide ini mirip dengan prinsip memilih variabel paling terbatas pada persoalan constraint satisfaction.

Heuristik tidak boleh menghapus kandidat secara sembarangan jika algoritma ingin tetap eksak. Kandidat boleh diurutkan, tetapi tidak boleh dibuang kecuali terdapat alasan bound yang sah. Jika kandidat dibuang karena hanya dianggap kurang baik secara heuristik, algoritma berubah menjadi heuristic search dan tidak lagi menjamin optimalitas.

G. Verifikasi Kebenaran Jadwal

Setelah sebuah jadwal dihasilkan, langkah verifikasi dilakukan dengan menghitung frekuensi kemunculan setiap pasangan. Jadwal valid jika setiap pasangan memiliki frekuensi minimal satu. Repetisi total dihitung sebagai jumlah kelebihan frekuensi di atas satu untuk seluruh pasangan. Verifikasi ini sederhana, tetapi penting karena kesalahan kecil pada pembentukan slot dapat menyebabkan satu peserta muncul di dua meeting pada slot yang sama atau menyebabkan pasangan tertentu tidak pernah bertemu.

Selain validitas cakupan, setiap slot juga diperiksa terhadap kendala konflik peserta. Untuk setiap slot, algoritma menghitung gabungan peserta dari seluruh meeting. Jika ukuran gabungan lebih kecil daripada jumlah total peserta yang tercantum dalam meeting-meeting slot tersebut, berarti ada peserta yang muncul lebih dari sekali dan slot tidak valid. Pemeriksaan ini dilakukan pada saat pembangkitan kandidat sehingga state yang tidak valid tidak pernah masuk ke pohon pencarian.

Validasi terakhir adalah membandingkan jumlah slot solusi dengan lower bound. Jika solusi memiliki jumlah slot sama dengan lower bound, maka optimalitas jumlah slot dapat langsung disimpulkan. Jika solusi lebih besar daripada lower bound, Branch and Bound harus terus mencari sampai semua cabang yang berpotensi mencapai jumlah slot lebih kecil atau sama telah dipangkas atau dieksplorasi.

TABLE III. PEMERIKSAAN VALIDITAS SOLUSI

Aspek	Kriteria Valid
Kapasitas meeting	Setiap meeting paling banyak b peserta.
Konflik slot	Peserta tidak muncul di dua meeting pada slot sama.
Cakupan pasangan	Setiap pasangan peserta muncul minimal sekali
Repetisi	Frekuensi di atas satu dihitung sebagai penalti.
Optimalitas slot	Dibuktikan jika semua cabang lebih

	baik telah dipangkas

VI. Implementasi dan Pengujian

A. Lingkungan Implementasi

Prototipe algoritma dibuat dalam Python. Setiap pasangan peserta direpresentasikan sebagai bit pada bitmask sehingga operasi gabungan dan selisih himpunan pasangan dapat dilakukan secara cepat. Seluruh kandidat meeting dibangkitkan dari kombinasi peserta berukuran dua sampai b. Kandidat slot waktu kemudian dibentuk secara rekursif dengan memastikan bahwa tidak ada dua meeting dalam slot yang memiliki peserta sama.

Untuk memperoleh solusi awal, digunakan greedy sederhana yang memilih konfigurasi slot dengan tambahan pasangan baru terbanyak. Solusi greedy ini menjadi upper bound awal bagi Branch and Bound. Dengan upper bound awal yang cukup baik, algoritma dapat memangkas lebih banyak cabang sejak awal pencarian.

B. Skenario Pengujian

Pengujian dilakukan pada beberapa kombinasi a dan b berukuran kecil sampai menengah. Ukuran yang terlalu besar tidak dipakai karena tujuan eksperimen bukan membuat sistem produksi, melainkan memperlihatkan cara kerja Branch and Bound, efektivitas bound, serta perbedaan fokus antara jumlah meeting dan jumlah slot waktu.

TABLE IV. Hasil Pengujian Prototipe

Kasus	a,b	LB slot	Hasil Branch and Bound
1	4,2	3	3 slot, 0 repetisi, 37 node
2	5,3	3	3 slot, 2 repetisi, 6.846 node
3	6,3	3	4 slot, 0 repetisi, 2,84 juta node
4	6,4	3	3 slot, 3 repetisi, 558.761 node

Angka node pada Table IV menunjukkan jumlah simpul pencarian yang dikunjungi pada prototipe. Nilai ini dapat berubah jika urutan kandidat atau teknik memoization diubah, tetapi tetap menunjukkan bahwa eksplorasi Branch and Bound meningkat cepat ketika struktur kasus menjadi lebih sulit.

C. Analisis Hasil

Hasil pada Table IV menunjukkan bahwa lower bound dapat langsung membuktikan optimalitas pada beberapa kasus. Misalnya, untuk a = 4 dan b = 2, setiap peserta hanya dapat bertemu satu orang baru per slot. Karena setiap peserta harus bertemu tiga peserta lain, minimal diperlukan tiga slot. Solusi greedy langsung mencapai tiga slot, sehingga Branch and Bound tidak perlu mengeksplorasi banyak cabang.

Kasus a = 6 dan b = 3 lebih menarik. Bound awal menyatakan bahwa tiga slot mungkin cukup secara perhitungan kasar, karena satu slot dapat mencakup enam pasangan dan total pasangan adalah 15. Namun kendala struktur meeting membuat tiga slot tidak cukup untuk menutup semua pasangan secara valid. Branch and Bound akhirnya menemukan solusi empat slot tanpa repetisi, seperti contoh pada Table II.

Pada a = 5 dan b = 3, solusi terbaik membutuhkan tiga slot dengan dua pengulangan pasangan. Hal ini wajar karena satu slot dapat mencakup paling banyak empat pasangan, sehingga tiga slot menyediakan 12 cakupan pasangan untuk menutup 10 pasangan unik. Akibatnya, sedikit pengulangan tidak dapat dihindari ketika seluruh pasangan harus tertutup dalam tiga slot.

Kasus a = 6 dan b = 4 memperlihatkan bahwa kapasitas lebih besar tidak selalu menghilangkan pengulangan. Tiga meeting berukuran empat dapat menutup semua pasangan dalam tiga slot, tetapi setiap meeting berukuran empat menghasilkan enam pasangan sehingga total cakupan menjadi 18 untuk 15 pasangan unik. Repetisi tiga pasangan tidak dapat dihindari pada solusi tiga slot yang ditemukan.

Secara umum, jumlah slot waktu adalah ukuran utama keberhasilan jadwal. Jumlah meeting total dapat berubah tergantung apakah algoritma memakai meeting besar atau beberapa meeting kecil, tetapi selama meeting tersebut dapat dijalankan paralel dan memenuhi kendala konflik peserta, pengaruhnya terhadap lama jadwal tidak sama dengan jumlah slot waktu. Oleh karena itu, objektif utama makalah ini tetap berupa minimisasi slot waktu.

D. Perbandingan dengan Greedy

Greedy cenderung cepat karena selalu memilih konfigurasi slot dengan tambahan pasangan baru terbanyak. Dalam beberapa kasus kecil, greedy menghasilkan jumlah slot yang sama dengan Branch and Bound. Namun greedy tidak menjamin optimalitas karena keputusan lokal dapat menyebabkan pasangan tertentu sulit ditutup pada slot berikutnya. Branch and Bound lebih mahal secara komputasi, tetapi mampu membuktikan optimalitas dengan bantuan bound.

Peran greedy dalam rancangan ini tetap penting sebagai pembentuk upper bound awal. Jika greedy menemukan solusi dengan jumlah slot yang sama dengan lower bound, Branch and Bound dapat segera menyimpulkan optimalitas objektif utama. Jika tidak, Branch and Bound tetap melakukan eksplorasi, tetapi dengan batas solusi awal yang lebih ketat.

VII. Kesimpulan dan Saran

Makalah ini telah memodelkan persoalan penjadwalan meeting berkapasitas terbatas dengan kendala konflik waktu peserta. Tujuan utama persoalan adalah meminimalkan jumlah slot waktu agar setiap pasangan peserta bertemu minimal sekali. Tujuan sekunder adalah meminimalkan pengulangan pasangan, karena jadwal ideal membuat setiap pasangan bertemu tepat satu kali.

Algoritma Branch and Bound diterapkan dengan branching berbasis konfigurasi slot waktu. Pendekatan ini lebih sesuai daripada branching per meeting karena kendala bahwa satu peserta hanya dapat mengikuti satu meeting dalam satu slot waktu langsung terpenuhi. Fungsi bound berbasis jumlah pasangan tersisa dan jumlah peserta yang belum ditemui oleh setiap peserta dapat memangkas cabang yang tidak mungkin mengalahkan solusi terbaik sementara.

Hasil pengujian menunjukkan bahwa Branch and Bound dapat menemukan dan membuktikan solusi optimal untuk ukuran kecil. Pada beberapa kasus, solusi greedy sudah mencapai lower bound sehingga optimalitas jumlah slot dapat disimpulkan dengan cepat. Namun, ketika lower bound masih longgar, Branch and Bound tetap memerlukan eksplorasi yang cukup besar. Hal ini menunjukkan bahwa kualitas bound sangat berpengaruh terhadap performa.

Pengembangan berikutnya dapat dilakukan dengan menambahkan bound yang lebih kuat, misalnya bound berbasis derajat graf pasangan yang belum tercakup atau formulasi integer programming. Selain itu, model dapat diperluas dengan preferensi peserta, ketersediaan waktu, prioritas pasangan tertentu, serta batas jumlah meeting maksimum pada setiap slot waktu.

REFERENSI

[1] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," Econometrica, vol. 28, no. 3, pp. 497-520, 1960.

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 4th ed. Cambridge, MA: MIT Press, 2022.

[3] R. M. Karp, "Reducibility among combinatorial problems," in Complexity of Computer Computations, R. E. Miller and J. W. Thatcher, Eds. New York: Plenum, 1972, pp. 85-103.

[4] CSPLib, "Problem 010: Social Golfers Problem." Accessed: 2026. [Online]. Available: https://www.csplib.org/Problems/prob010/

[5] D. Schmand, M. Schroeder, and L. Vargas Koch, "A greedy algorithm for the social golfer and the Oberwolfach problem," arXiv:2007.10704, 2020.

[6] J. Kleinberg and E. Tardos, Algorithm Design. Boston, MA: Pearson, 2006.

[7] C. H. Papadimitriou and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity. Mineola, NY: Dover Publications, 1998.

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 Juni 2026

Muhammad Faiz Alfikrona (18223009)