

Pemodelan Strategi Bluffing pada Permainan Texas Hold'em Poker menggunakan Graf Berbobot dan Algoritma Pengambilan Keputusan

Al Farabi - 13524086

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jalan Ganesha 10 Bandung

E-mail: alfarabi0642@gmail.com, 13524086@std.stei.itb.ac.id

Abstrak—Texas Hold'em Poker adalah permainan kartu komunitas yang menggunakan satu set kartu 52 kartu. Setiap pemain menerima dua kartu pribadi (“hole cards”) dan berbagi lima kartu komunitas yang dibagikan dalam beberapa tahap (flop, turn, river). Meskipun struktur taruhan dan taktik psikologis sering dianggap kunci, pengambilan keputusan optimal dapat ditingkatkan secara signifikan dengan menerapkan probabilitas kombinatorial atas susunan kartu, bukan semata-mata mengandalkan intuisi atau besar taruhan. Karena komposisi dek diketahui dan terbatas, pemain dapat menghitung peluang menyelesaikan undian (misalnya flush atau straight) atau meningkatkan kekuatan tangan pada setiap tahap. Dengan memetakan setiap titik keputusan sebagai graf berbobot dari kemungkinan hasil kartu, dapat ditunjukkan bagaimana analisis kombinatorial sederhana dapat mengidentifikasi aksi yang secara matematis optimal, bahkan mendukung strategi bluffing, tanpa harus terjebak dalam simulasi taruhan kompleks.

Keywords—Texas Hold'em Poker, bluffing, algoritma DFS, graf berbobot

I. PENDAHULUAN

Permainan poker merupakan salah satu permainan kartu yang sangat populer di seluruh dunia. Salah satu aspek terpenting dalam poker adalah strategi bluffing, yaitu upaya untuk menipu lawan dengan memberikan persepsi yang salah mengenai kartu yang dimiliki [1]. Strategi bluffing tidak hanya tentang persoalan keberuntungan, tetapi juga melibatkan proses pengambilan keputusan dan analisis situasi yang kompleks secara dinamis.

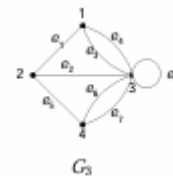
Seiring berkembangnya teknologi dan metode komputasi, penelitian mengenai pemodelan strategi bluffing semakin berkembang, khususnya dengan pendekatan matematis dan algoritmik. Makalah ini akan mencoba memodelkan strategi bluffing pada permainan Texas Hold'em Poker dengan memanfaatkan graf berbobot dan algoritma pengambilan keputusan. Dengan menggabungkan graf berbobot dan algoritma pengambilan keputusan, strategi bluffing dapat dioptimalkan dan disesuaikan dengan kondisi permainan yang dinamis.

II. LANDASAN TEORI

A. Graf

1) Definisi

Graf digunakan untuk merepresentasikan hubungan antara objek-objek diskrit.



Gambar 1. Graf semu

(Sumber:

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf>)

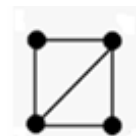
Graf $G = (V, E)$ terdiri dari himpunan simpul (vertices) $V = \{v_1, v_2, \dots, v_n\}$ dan himpunan sisi (edges) $E = \{e_1, e_2, \dots, e_m\}$ yang menghubungkan sepasang simpul V .

2) Jenis-jenis graf

Berdasarkan ada tidaknya gelang atau sisi ganda pada graf, graf dapat digolongkan menjadi dua jenis:

- Graf sederhana (*simple graph*)

Graf yang tidak mengandung gelang maupun sisi ganda.



Gambar 2. Graf Sederhana

(Sumber:

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf>)

- Graf tak-sederhana (*unsimple-graph*)

Graf yang mengandung sisi ganda atau gelang.



Gambar 3. Graf tak-sederhana

(Sumber:

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf>)

3) Orientasi arah graf

Berdasarkan orientasi arah pada sisi, graf dibedakan menjadi 2 jenis:

- Graf tak-berarah (*undirected graph*)
Graf yang sisinya tidak mempunyai orientasi arah.



Gambar 4. Graf tak-berarah

(Sumber:

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf>)

- Graf berarah (*directed graph*)
Graf yang setiap sisinya diberikan orientasi arah.



Gambar 5. Graf berarah

(Sumber:

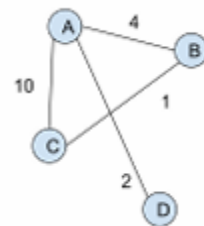
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf>)

4) Terminologi graf

- Ketetanggaan (*Adjacent*)
Dua buah simpul dikatakan bertetangga bila keduanya terhubung langsung oleh satu sisi yang sama.
- Bersisian (*Incidency*)
Untuk suatu sisi $e = (v_j, v_k)$ dikatakan e bersisian dengan simpul v_j dan/atau e bersisian dengan simpul v_k .
- Simpul terpencil (*Isolated vertex*)
Simpul yang tidak mempunyai sisi yang bersisian dengannya.
- Graf kosong (*Empty graph*)
Graf yang himpunan sisinya kosong.
- Derajat (*Degree*)
Jumlah sisi yang bersisian dengan suatu simpul.
- Lintasan (*Path*)
Lintasan yang panjangnya n dari simpul awal v_0 ke simpul tujuan v_n pada graf G ialah barisan berselang-seling simpul-simpul dan sisi-sisi yang

terbentuk $v_0, e_1, v_1, e_2, v_2, \dots, v_{n-1}, e_n, v_n$ sedemikian sehingga $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2)$, $\dots$, $e_n = (v_{n-1}, v_n)$ adalah sisi-sisi dari graf G . Panjang lintasan adalah jumlah sisi dalam lintasan tersebut.

- Sirkuit (*Circuit*)
Siklus atau sirkuit adalah lintasan yang berawal dan berakhir pada simpul yang sama. Panjang sirkuit adalah jumlah sisi pada sirkuit tersebut.
- Keterhubungan (*Connected*)
Dua buah simpul v_1 dan v_2 dikatakan terhubung apabila ada suatu lintasan yang menghubungkan dari v_1 ke v_2 . Graf G dikatakan terhubung (*connected graph*) apabila setiap pasang simpul v_i dan v_j pada G memiliki lintasan yang menghubungkan keduanya.
- Upagraf (*Upagraph*)
Misalkan $G = (V, E)$ adalah sebuah graf. $G_1 = (V_1, E_1)$ adalah sebuah upagraf dari G jika $V_1 \subseteq V$ dan $E_1 \subseteq E$.
- Upagraf merentang (*Spanning subgraph*)
Upagraf $G_1 = (V_1, E_1)$ dari $G = (V, E)$ dikatakan upagraf merentang jika G_1 mengandung semua simpul dari G .
- Cut-set
Cut-set dari graf terhubung G adalah himpunan sisi yang bila dibuang dari G menyebabkan G tidak terhubung.
- Graf berbobot
Graf dikatakan berbobot apabila setiap sisi E diberikan nilai bobot w . bobot ini dapat merepresentasikan biaya, jarak, atau payoff pada permasalahan pengambilan keputusan.



Gambar 6. Graf berbobot

(Sumber:

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf>)

B. Lintasan Optimal

- Bobot jalur

Bobot jalur adalah jumlah bobot sisi sepanjang jalur (*path*).

- Jalur terpendek

Jalur terpendek pada graf berbobot adalah jalur (*path*) yang memiliki bobot terkecil atau minimum.

- Jalur maksimum

Jalur maksimum pada graf berbobot adalah jalur (*path*) dengan bobot paling maksimal atau maksimum.

C. Teori dasar permainan

- Struktur permainan

Texas Hold'em Poker menggunakan satu set kartu remi standar sebanyak 52 lembar, yang terbagi dalam empat jenis: Wajik, Hati, Keriting, dan Sekop. Setiap jenis memiliki satu kartu As, angka 2 hingga 10, serta tiga kartu wajah (Jack, Queen, King). Pada Hold'em, kartu As dapat dihitung sebagai 1 atau 14; Jack bernilai 11, Queen bernilai 12, dan King bernilai 13.

Permainan dimulai dengan setiap pemain menerima dua kartu tertutup (disebut "*hole cards*"). Setelah itu, diadakan ronde taruhan pertama. Kemudian, tiga kartu komunitas dibuka bersama-sama ("*flop*"), diikuti oleh ronde taruhan kedua. Kartu komunitas keempat ("*turn*") dibuka dan diikuti ronde taruhan ketiga, sebelum akhirnya kartu komunitas kelima ("*river*") dibuka untuk ronde taruhan terakhir. Pada tahap showdown, setiap pemain menyusun *hand-rank* untuk menentukan pemenang. *Hand-rank* merupakan peringkat kombinasi yang dapat terbentuk dari lima lembar kartu, dan menjadi tolak ukur kemenangan dalam permainan Texas Hold'em Poker.[3]

- Nilai kombinasi kartu

Pemenang ditentukan oleh peringkat kombinasi lima kartu (*hand-rank*) tertinggi. Berikut urutan *hand-rank* dari tertinggi hingga terendah [3]:

1. Royal Flush: Lima kartu berurutan dari 10 hingga As, semuanya sejenis. Peringkat teratas, otomatis menang; jika dua pemain memilikinya, pot dibagi rata.
2. Straight Flush: Lima kartu berurutan dengan jenis sama, mulai dari A-5 (As bernilai 1) hingga 9-K (As bernilai 14).
3. Four of a Kind: Empat kartu sama nilai plus satu kartu tambahan sebagai penentu jika seri.
4. Full House: Tiga kartu sama nilai ditambah sepasang kartu lain sejenis.
5. Flush: Lima kartu sejenis, tanpa urutan; penilaian dibuat berdasarkan urutan nilai kartu tertinggi.
6. Straight: Lima kartu berurutan tanpa harus sejenis; As dapat bernilai 1 atau 14 sesuai urutan.
7. Three of a Kind: Tiga kartu sama nilai, ditambah dua kartu tunggal sebagai penentu jika seri.
8. Two Pair: Dua pasang kartu sama nilai dan satu kartu tambahan untuk menentukan seri.
9. Pair: Sepasang kartu sama nilai ditambah tiga kartu tunggal sebagai penentu seri.
10. High Card: Jika tidak ada kombinasi di atas, lima kartu tertinggi dijadikan penentu, dibandingkan dari yang terbesar.

- Alur Dasar Permainan Texas Hold'em Poker:

Alur dasar permainan Texas Hold'em Poker yang normal adalah seperti ini [3]:

1. Setiap pemain menerima dua kartu tertutup (*hole cards*) secara rahasia.
2. *Pre-flop*: Ronde taruhan pertama dimulai setelah pembagian *hole cards*.
3. *Flop*: Tiga kartu komunitas pertama dibuka di atas meja, dilanjutkan ronde taruhan kedua.
4. *Turn*: Kartu komunitas keempat dibagikan terbuka, diikuti ronde taruhan ketiga.
5. *River*: Kartu komunitas kelima dibuka, kemudian ronde taruhan keempat dan terakhir dilakukan.
6. *Showdown*: Jika lebih dari satu pemain tetap bertaruh setelah *river*, para pemain menunjukkan *hole cards* mereka dan menentukan kombinasi lima kartu terbaik.
7. Pemain dengan *hand-rank* tertinggi memenangkan pot; jika terjadi seri pada *hand-rank* tertinggi, pembanding kartu penentu (*kickers*) atau aturan pembagian pot digunakan.

Permainan dapat usai sebelum *showdown* jika hanya tersisa satu pemain yang masih bertaruh, pemain tersebut langsung dinyatakan sebagai pemenang.

D. Konsep matematika dalam poker

- Probabilitas dan statistik

Probabilitas dalam poker merupakan fondasi pengambilan keputusan strategis. Perhitungan probabilitas melibatkan distribusi hipergeometrik untuk menentukan *equity* (persentase kemenangan) berdasarkan *outs* (kartu yang dapat memperbaiki kombinasi *hand*). Distribusi ini memodelkan peluang munculnya kartu tertentu dari sisa dek yang belum terlihat [4].

TABLE I. WINNING PERCENTAGE OF 5 HIGHEST CARDS

Hand	Winning Percentage
AA	86.1%
KK	74.6%
AK	68.6%
QQ	68.5%
AK	67.0%

- Expected value

Expected value (EV) mengkuantifikasi keuntungan/kerugian rata-rata dari suatu aksi dalam jangka panjang. Formula dasar EV [6]:

$$ev = \text{prob_win} \times \text{amount won} - (\text{problose} \times \text{amount_lost}) \quad (1)$$

Dalam poker, keputusan optimal selalu memaksimalkan EV positif jangka panjang [7]. Pemain ahli menggunakan konsep EV untuk membandingkan aksi *fold*, *call*, atau *raise* dalam situasi kompleks.

- *Pot odds* dan *implied odds*
Pot odds adalah rasio antara biaya *call* terhadap total pot setelah *call*. Formula *pot odds*:

$$\text{pot odds} = \text{jumlah call} / (\text{pot now} + \text{jumlah call}) \quad (2)$$

Implied odds memperhitungkan potensi keuntungan tambahan di *street* berikutnya jika *call*.

E. Pohon Keputusan

Pohon keputusan adalah metode klasifikasi yang mengadopsi pendekatan *divide-and-conquer* [3]. Algoritma ini bekerja dari akar pohon, memilih atribut terbaik untuk membagi data menjadi subset, kemudian memproses setiap subset secara rekursif hingga membentuk aturan klasifikasi.

Struktur decision tree terdiri atas tiga jenis node [3]:

1. *Root Node*: Titik awal yang tidak memiliki input, namun dapat memiliki satu atau lebih cabang output[3].
2. *Internal Node*: Titik percabangan yang menerima satu input dan memiliki minimal dua cabang output[3].
3. *Leaf Node (Terminal Node)*: Titik akhir yang hanya memiliki satu input dan tidak memiliki cabang lanjutan mewakili kelas atau keputusan akhir[3].

Pada setiap internal node, keputusan diambil berdasarkan kondisi *if-then*, tanpa memerlukan parameter eksternal. Keunggulan *decision tree* meliputi struktur yang mudah diinterpretasi, kemampuan menangani atribut multi-tipe, serta toleransi terhadap data hilang (*missing*) atau *noise*.

Dalam konteks Texas Hold'em Poker, *decision tree* telah diterapkan pada sistem seperti DeepStack dan Libratus, serta penelitian lain yang mengembangkan strategi optimal untuk dimainkan manusia [3].

F. Konsep Bluffing dalam Teori Permainan

Bluffing dalam poker merupakan masalah teori permainan klasik di mana pemain menggunakan strategi campuran untuk memaksimalkan keuntungan ekspektasi (*expected value*). *Bluffing* bukan sekadar taktik psikologis melainkan solusi matematis optimal untuk situasi ketidakpastian informasi [1]. Secara sederhana dapat dikatakan, pemain yang menggunakan strategi *bluffing* berpura-pura memiliki kartu yang kuat dengan tujuan membuat lawan *fold*.

G. Algoritma Pengambilan Keputusan

- Expectimax algorithm
Pada algoritma *expectimax*, ketika kita mengevaluasi simpul lawan, kita akan mengkalkulasi seluruh gerakan yang mungkin, yang memiliki bobot berdasarkan kemungkinan terjadinya. Dalam kata lain, kita akan mengkalkulasikan *expected value* (EV) berdasarkan bobot dari setiap kemungkinan [2]. Contoh *Pseudocode expectimax algorithm* menurut Zaky (2014):

```
function expectimax(node, depth, turn)
  if depth = 0 or node is a terminal node
    return the heuristic value of node
  if turn == player1
```

```
    bestValue := -∞
    for each child of node
      val := expectimax(child, depth-1,
        player2)
      bestValue := max(bestValue, val)
    return bestValue
  else
    expectedValue := 0
    for each child of node
      val := expectimax(child, depth-1,
        player1)
      expectedValue += Probability[child] *
        val
    return expectedValue

(* Initial call *)
expectimax(root, depth, player1)
```

- Regret minimization

Regret minimization adalah kerangka algoritmik untuk mengoptimalkan strategi dalam permainan sekuensial. Diberikan urutan profil strategi $\sigma^1, \sigma^2, \dots, \sigma^T$, *external regret* untuk pemain i didefinisikan sebagai [11]:

$$R_i^T = \max_{\sigma_i^* \in \Sigma_i} \sum_{t=1}^T (u_i(\sigma_i^*, \sigma_{-i}^t) - u_i(\sigma^t))$$

Gambar 7. Equation of external regret

di mana:

Σ_i : ruang strategi pemain i

u_i : Fungsi utilitas pemain i

σ_{-i}^t : Strategi lawan pada iterasi t

Regret R_i^T mengkuantifikasi *utility* yang “hilang” karena tidak menggunakan strategi tunggal terbaik (σ_i^*) sepanjang T iterasi [8].

III. PEMODELAN, ANALISIS, DAN PEMBAHASAN STRATEGI BLUFFING

Pemodelan strategi *bluffing* dimulai dengan membuat asumsi terkait profil atau karakter bermain lawan. Profil lawan yang dipakai untuk pemodelan ini ada 3 jenis, yaitu *Tight Passive*, *Loose Aggressive*, dan *Scaredy Cat*. Masing-masing dari profil lawan memiliki kemungkinan aksi (*fold*, *call*, *raise*) yang berbeda-beda untuk merepresentasikan karakter bermain lawan. Berikut klasifikasi profil awal lawan yang digunakan pada pemodelan:

- *Tight Passive*:

TABLE II. ASUMSI KEMUNGKINAN AKSI TIGHT PASSIVE

Fold	40%
Call	40%
Raise	20%

- *Loose Aggressive:*

TABLE III. ASUMSI KEMUNGKINAN AKSI *LOOSE AGGRESSIVE*

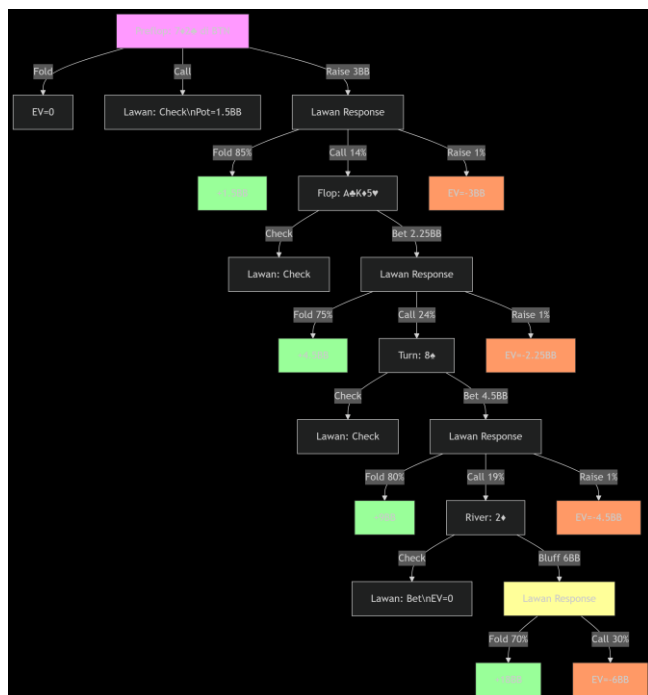
Fold	10%
Call	30%
Raise	60%

- *Scaredy Cat*

TABLE IV. ASUMSI KEMUNGKINAN AKSI *SCAREDY CAT*

Fold	85%
Call	14%
Raise	1%

Kita menggunakan data simulasi profil lawan, dan melakukan kalkulasi ev pada setiap *edge* di dalam *decision tree*. Di mana *edge* merepresentasikan aksi yang dilakukan oleh pemain, dengan bobot EV, dan *vertex* merepresentasikan situasi atau *stage* dalam game. Berikut adalah model *decision tree* pada strategi *bluffing* untuk situasi *heads-up all-stage* (dari *preflop* hingga *river*), dengan profil lawan *Scaredy-Cat* pada posisi BB (*Big Blind*) 1 dan pemain pada posisi BTN, yang disesuaikan dengan aksi historis lawan secara dinamis.



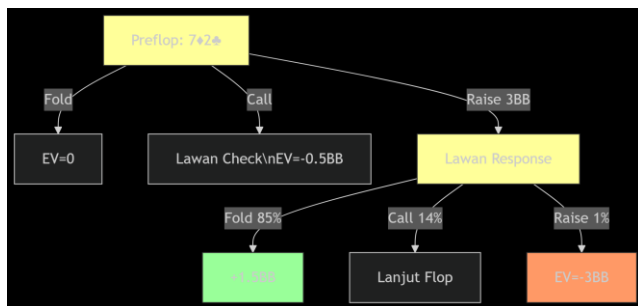
Gambar 8. Model pohon keputusan dari *flop* hingga *showdown*

EV (*expected value*) dihitung berdasarkan aksi yang pemain atau lawan pilih. Formula $EV = \text{kemungkinan_aksi_lawan} * (\text{pot} - \text{call_amount})$. Berikut kode *python* sederhana untuk perhitungan ev:

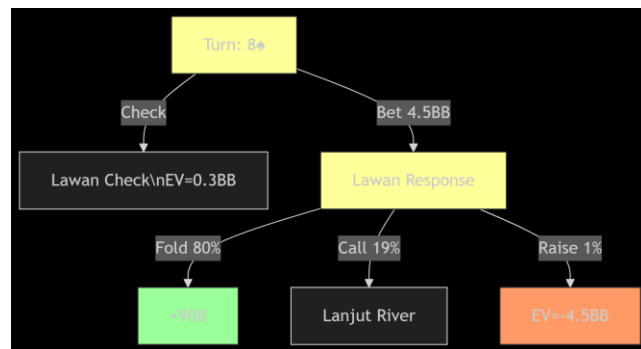
```
def calculate_ev(self, action, bet_size=0):
    if action == ActionType.FOLD:
        return 0
    resp_probs = self.opponent_response(action)
    ev = 0
    for resp, prob in resp_probs.items():
        if resp == "FOLD":
            ev += prob * self.pot
        elif resp == "CALL":
            if self.stage == GameStage.RIVER:
                equity = self.estimate_equity()
                win_amount = self.pot + 2 * bet_size
                ev += prob * (equity * win_amount - (1 - equity) * bet_size)
            else:
                penalty = 1.3 if self.opponent_type == PlayerType.LOOSE_AGGRESSIVE else 1.0
                ev += prob * (-bet_size * penalty)
        elif resp == "RAISE":
            penalty = 1.5 if self.opponent_type == PlayerType.LOOSE_AGGRESSIVE else 1.0
            ev += prob * (-bet_size * penalty)
        elif resp == "BET":
            ev += prob * 0
    return ev
```

A. Pemodelan Preflop

Pemodelan *preflop* menggunakan asumsi awal, dan pemain mendapatkan kartu 7 wajik dan 2 keriting. Jika kita memilih melakukan *fold* kita tidak mendapatkan keuntungan apa-apa (tidak melakukan *bluffing*), atau jika kita *call* atau *raise* akan didapat EV untuk setiap kemungkinan aksi lawan. Jika tidak ada salah satu dari pemain atau lawan melakukan *fold*, maka game *stage* beralih ke *stage flop*.



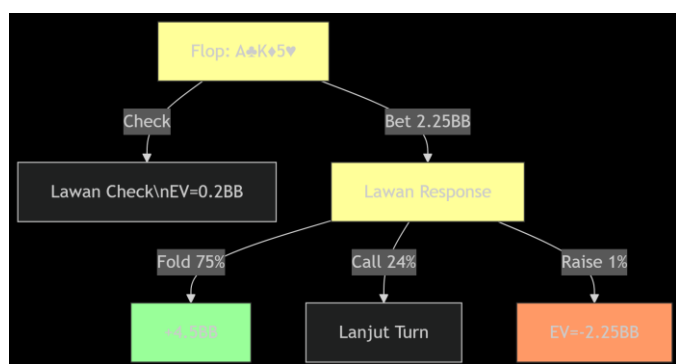
Gambar 9. Model pohon keputusan *preflop*



Gambar 11. Model pohon keputusan *turn*

B. Pemodelan Flop

Pemodelan *flop* dengan asumsi awal, dan *flop* diperlihatkan sebagai A keriting, K wajik, dan 5 Hati. Pada posisi ini, pemain memiliki equity yang cukup rendah, dikarenakan tidak memiliki kombinasi *hand* yang tinggi. Pada posisi ini juga, pemain dapat memilih *check*, *fold*, ataupun *raise*. Dengan EV tertinggi ada pada pilihan *raise*, berdasarkan formula EV yang dibuat sebelumnya. Jika tidak ada salah satu dari pemain atau lawan yang melakukan *fold*, maka *game* berlanjut ke *stage turn*.



Gambar 9. Model pohon keputusan *flop*

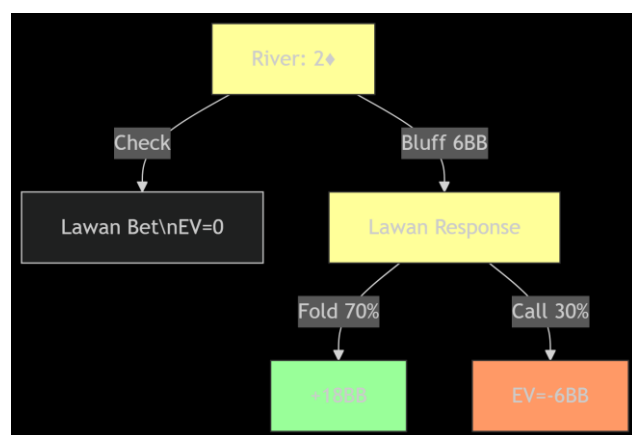
C. Pemodelan Turn

Pemodelan *turn* dengan asumsi awal, dan kartu *turn* adalah 6 wajik. Pada posisi ini, pemain memiliki equity yang sangat rendah, dikarenakan tidak memiliki kombinasi *hand* yang tinggi dan dapat diasumsikan kartu lawan memiliki equity yang tinggi, karena lawan tidak memilih *fold* pada fase *preflop*. Pada posisi ini juga, pemain dapat memilih *check*, *fold*, ataupun *raise*. Dengan EV tertinggi ada pada pilihan *raise*, berdasarkan formula EV yang dibuat sebelumnya. Jika tidak ada salah satu dari pemain atau lawan yang melakukan *fold*, maka *game* berlanjut ke *stage turn*.

D. Pemodelan River

Pemodelan *river* dengan situasi lanjutan dari *stage turn*, dan kartu *river* adalah J sekop. Pada posisi ini, pemain memiliki equity yang hampir 0, dikarenakan tidak memiliki kombinasi sama sekali pada *hand* yang dimiliki dan dapat diasumsikan kartu lawan memiliki equity yang sangat tinggi dibandingkan dengan kita, karena lawan tidak memilih *fold* sejak *game* awal dengan kemungkinan *fold* tinggi. Pada posisi ini juga, pemain dapat memilih *check*, *fold*, ataupun *raise*.

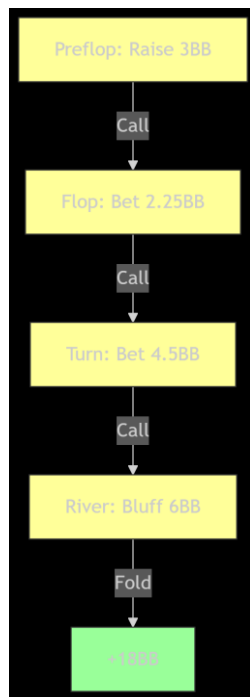
Namun, melakukan *fold* adalah pilihan yang tidak logis, karena sudah terlalu banyak *chip* yang diinvestasikan, maka pemain dapat mengeliminasi pilihan itu. Melakukan *check* juga tidak akan mendapati pemain EV yang lebih dari EV yang didapat jika memilih aksi *raise*. Maka, aksi paling optimal adalah melakukan *raise*, dengan hasil prediksi (berdasarkan asumsi) akhir, kemungkinan lawan akan melakukan *fold* adalah sebesar 70%. Namun, besaran dari *raise* pun berpengaruh pada pengambilan keputusan lawan, jika pemain memilih untuk *raise* sebesar *All-In*, kemungkinan pemain berhasil melakukan *bluff* semakin tinggi. *Bluff* pemain berhasil jika lawan memilih aksi *fold* pada akhir permainan.



Gambar 9. Model pohon keputusan *river*

E. Model Optimal

Model optimal adalah representasi situasi optimal penerapan strategi *bluffing*:



Gambar 9. Model pohon keputusan strategi optimal *bluffing*

F. Hasil Kode Python

Penulis melakukan simulasi sebanyak 500 kali per profil lawan untuk pemodelan yang lebih optimal dalam menunjukkan EV setiap simulasi. Hasil *output* kodenya adalah sebagai berikut:

```
===== CONTOH 4: ANALISIS EV STRATEGI BLUFF (500 SIMULASI) =====
Tipe Lawan      Avg Total EV (BB)      Avg EV/Decision (BB)
-----
TIGHT_PASSIVE   3.03                   0.92
LOOSE_AGGRESSIVE -64.17                 -16.52
SCAREDY_CAT     2.25                   1.21

KESIMPULAN STRATEGI BERDASARKAN EV:
- Melawan TIGHT_PASSIVE: EV positif (0.92 BB/decision)
- Melawan LOOSE_AGGRESSIVE: EV negatif (-16.52 BB/decision)
- Melawan SCAREDY_CAT: EV positif (1.21 BB/decision)
```

Gambar 11. Hasil kode *python*
(Sumber: kode *python* penulis)

Pada pemodelan melawan profil lawan *Tight Passive*, lawan pasif memberikan EV yang tinggi untuk hero melakukan *bluffing*. Hal ini dikarenakan lawan cukup sering *fold*. Namun, pada pemodelan melawan profil *Loose Aggressive*, aksi *bluffing* hero hampir selalu gagal, dikarenakan lawan agresif. Hasil negatif yang signifikan menunjukkan bahwa melawan profil lawan yang agresif, strategi *bluffing* sederhana tanpa adaptabilitas yang cukup, justru merugikan. Di sisi lain, pada pemodelan melawan profil *Scaredy Cat*, lawan yang mudah “ketakutan” ini sangat rentan terhadap *bluff*. Oleh karena itu, EV cukup tinggi ketika melakukan *bluff* melawan profil *Scaredy Cat*. Hasil ini selaras dengan temuan *Theory of Game* bahwa *bluff* hanya menguntungkan bila lawan memiliki kecenderungan *fold* cukup besar.

IV. KESIMPULAN

Makalah ini memodelkan strategi *bluffing* pada permainan Texas Hold'em Poker melalui representasi graf berbobot dan algoritma pengambilan keputusan. Setiap simpul (*vertex*) mencerminkan situasi permainan mulai dari *preflop*, *flop*, *turn*, hingga *river* sementara bobot pada sisi (*edge*) merepresentasikan *Expected Value* (EV) untuk aksi *fold*, *call*, atau *raise*. Secara matematis, pohon keputusan berdimensi terbatas (*finite-horizon*) ini memadai karena jumlah *i* yang berhingga dan bobot EV pada setiap cabang dapat dihitung dari probabilitas respons lawan dan *payoff* serta algoritma menjamin pemilihan jalur dengan EV maksimal, yang setara dengan jalur terberat dalam graf berbobot.

Simulasi terhadap tiga profil lawan *Tight Passive*, *Loose Aggressive*, dan *Scaredy Cat* menghasilkan pola keputusan EV-optimal yang konsisten. Pada tahap *preflop* dan *flop*, model merekomendasikan aksi *check* atau *call* ketika EV dari *raise* belum menembus threshold positif. Di tahap *turn*, model mulai memprioritaskan *raise*, terutama melawan profil *Loose Aggressive*, karena peningkatan EV yang signifikan. Sedangkan di tahap *river*, agresivitas melalui *raise* memberikan EV tertinggi terhadap profil *Scaredy Cat* dengan probabilitas *fold* lawan mencapai sekitar 70 persen.

Penelitian ini memiliki keterbatasan pada asumsi profil lawan yang statis dengan probabilitas respons tetap, sehingga tidak mencerminkan adaptasi lawan selama permainan. Selain itu, perhitungan EV masih sederhana tanpa mempertimbangkan *pot odds* jangka pendek maupun *implied odds* jangka panjang. Pemodelan ini oleh karena itu belum sepenuhnya mencerminkan kompleksitas dinamika taruhan dan strategi lawan di dunia nyata.

Untuk meningkatkan akurasi model, disarankan mengembangkan algoritma *opponent modelling* dinamis yang memperbarui profil lawan berdasarkan riwayat aksi dan pola taruhan. Integrasi *pot odds* dan *implied odds* di setiap tahap dapat memberikan estimasi EV yang lebih realistis. Selanjutnya, eksperimen empiris bersama pemain manusia perlu dilakukan untuk mengevaluasi kinerja model ini dibandingkan strategi konvensional atau AI poker komersial. Terakhir, penerapan pendekatan graf-EV pada varian poker lain, seperti Omaha, ataupun skenario *multi-player* dapat memperluas generalisasi dan aplikasi penelitian ini.

Pendekatan graf berbobot berpadu EV sederhana ini diharapkan menjadi fondasi bagi strategi *bluffing* optimal dan pengembangan AI poker berbasis data yang lebih kompleks.

VIDEO LINK AT YOUTUBE

https://youtu.be/iASvdPQlFE?si=zbO_WnbQQAo9FFbU

IMPLEMENTASI KODE PYTHON PEMODELAN DI GITHUB

https://github.com/alfarabi0642/Bluff_Expectimax_Algorithm_Modelling.git

UCAPAN TERIMA KASIH

Pada kesempatan ini, penulis ingin mengucapkan terima kasih kepada Tuhan Yang Maha Esa sehingga penulis mampu menyelesaikan makalah ini. Penulis juga ingin mengucapkan terima kasih kepada orang tua, keluarga, dan rekan-rekan penulis atas dukungan yang telah diberikan kepada penulis. Penulis juga ingin menyampaikan rasa terima kasih penulis kepada Pak Arrival Dwi Sentosa dan Pak Rinaldi Munir selaku dosen mata kuliah matematika diskrit yang telah membimbing penulis selama masa perkuliahan.

REFERENSI

- [1] L. Friedman, "Optimal bluffing strategies in poker," *Management Science* 17(12), 2016, pp. 764.
- [2] A. Zaky, "Minimax and expectimax algorithm to solve 2048," Makalah IF2211 Strategi Algoritma – Sem. II Tahun 2013/2014, Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, 2014, unpublished.
- [3] P. Indriyanti, M. F. Hismawan, dan Mujiono, "Penerapan decision tree dalam pengambilan keputusan untuk pemain texas hold'em poker," *J. Ilm. FIFO*, vol. 12, no. 2, 2020, pp. 167-178.
- [4] B. Chen and J. Ankenman, "The Mathematics of Poker," Pittsburgh, PA: Conjelco Publishing, 2006.
- [5] [ganti] persi dalam permainan kartu
- [6] D. Sklansky, "The Theory of Poker," Vol. 4. Henderson, NV: Two Plus Two Publishing, 1999.

- [7] J. Von Neumann and O. Morgenstern, "Theory of Games and Economic Behavior," Princeton, NJ: Princeton Univ. Press, 1944.
- [8] R. Gibson, "Regret Minimization in Games and the Development of Champion Multiplayer Computer Poker-playing Agents," Ph.D. dissertation, Dept. of Computing Science, University of Alberta, Edmonton, AB, Canada, Spring 2014.
- [9] R. Munir, "Graf (bag.1)." Accessed: June. 18, 2025. [Online]. Available: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf>

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, 1 Juni 2025



Al Farabi,
13524086