PERTEMUAN KE-2

Masalah dan Ruang Keadaan

Kecerdasan Buatan — Teknik Informatika

Dosen: **Dr. Sayuti Rahman** | Fakultas Teknik

Sistem Kecerdasan Buatan

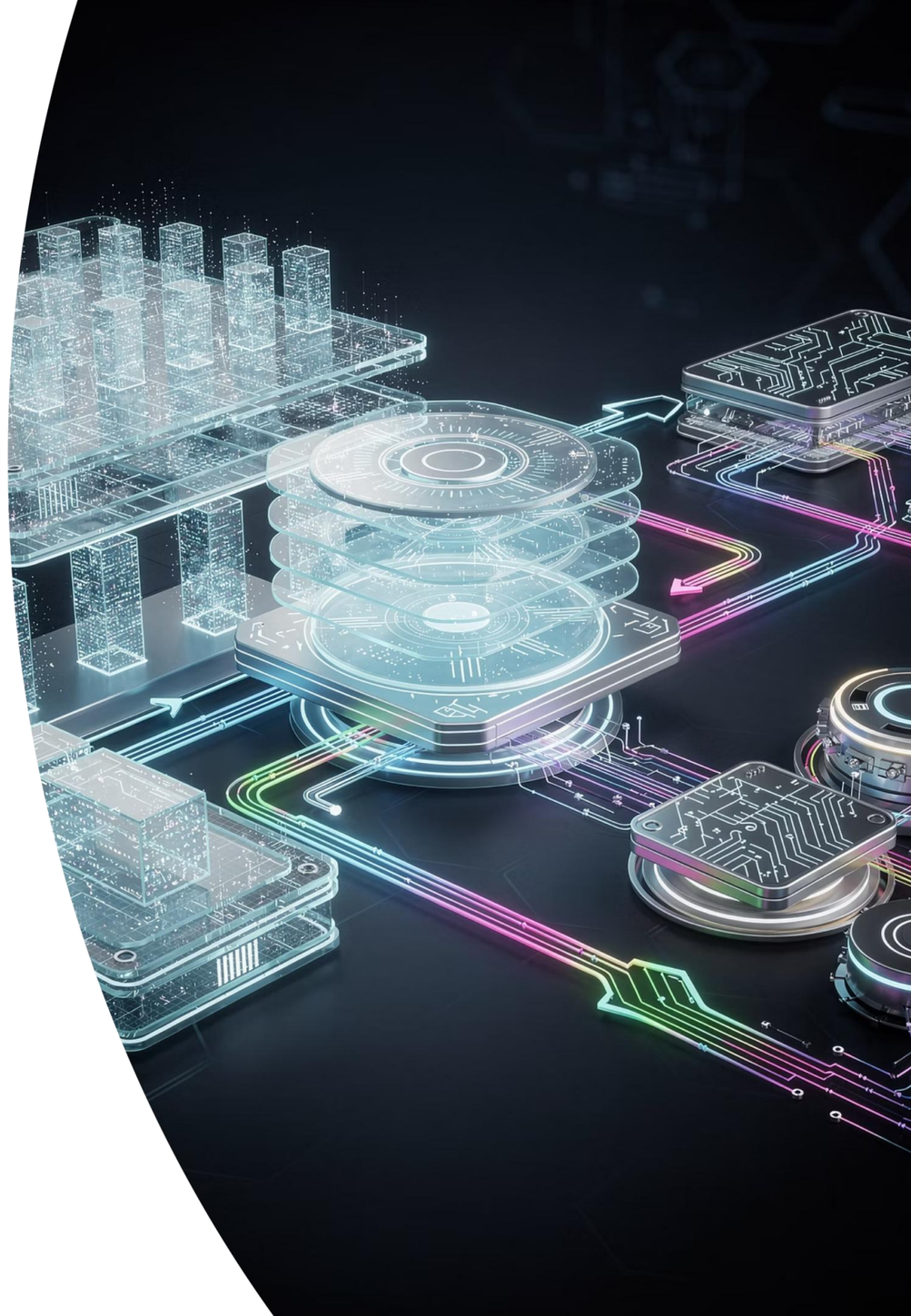
Untuk membangun aplikasi kecerdasan buatan, terdapat **2 bagian utama** yang sangat dibutuhkan agar sistem dapat berjalan secara efektif dan cerdas:

Basis Pengetahuan *(Knowledge Base)*

Berisi fakta-fakta, teori, pemikiran, dan hubungan antara satu dengan lainnya. Merupakan fondasi informasi yang digunakan sistem untuk memahami dunia.

Motor Inferensi *(Inference Engine)*

Yaitu kemampuan menarik kesimpulan berdasarkan pengalaman dan pengetahuan yang tersimpan. Motor inferensi mengolah basis pengetahuan untuk menghasilkan keputusan.



Masalah dalam Kecerdasan Buatan

Untuk membangun sistem yang mampu menyelesaikan masalah menggunakan AI, diperlukan pendekatan sistematis yang mencakup empat langkah utama berikut:

1

Definisikan Masalah

Mencakup spesifikasi yang tepat mengenai **keadaan awal** dan **solusi yang diharapkan** dari sistem.

2

Analisis Masalah

Menganalisis masalah tersebut serta mencari beberapa **teknik penyelesaian masalah** yang paling sesuai.

3

Representasi Pengetahuan

Merepresentasikan pengetahuan yang diperlukan agar sistem dapat memahami dan memproses masalah dengan benar.

4

Pilih Teknik Terbaik

Memilih teknik penyelesaian masalah yang **paling optimal** berdasarkan analisis dan ketersediaan sumber daya.

Mendefinisikan Suatu Masalah

Dalam pendekatan AI berbasis ruang keadaan, mendefinisikan masalah secara formal adalah langkah krusial. Setiap masalah harus diurai menjadi komponen-komponen berikut:

Buat *State Space* (Ruang Masalah)

Definisikan seluruh kemungkinan keadaan yang dapat terjadi dalam masalah tersebut. Ruang masalah adalah fondasi dari seluruh proses pencarian solusi.

Tentukan *Initial State* (Keadaan Awal)

Tentukan kondisi awal sistem sebelum proses pencarian solusi dimulai. Ini adalah titik tolak dari seluruh eksplorasi.

Tentukan *Goal State* (Keadaan Tujuan)

Tentukan kondisi akhir yang ingin dicapai. Sistem akan terus mencari jalur hingga keadaan tujuan ini terpenuhi.

Tentukan *Rule* (Aturan)

Tetapkan aturan-aturan yang mengatur transisi dari satu keadaan ke keadaan lainnya secara legal dan valid.

Masalah sebagai Ruang Keadaan: Permainan Catur

Permainan catur merupakan contoh klasik representasi masalah dalam ruang keadaan. Berikut komponen ruang keadaannya:

1. Posisi Awal

Posisi awal setiap permainan catur selalu sama: semua bidak diletakkan di atas papan dalam 2 posisi, yaitu **kubu putih** dan **kubu hitam**. Ini adalah *initial state* yang konsisten dan terstandarisasi.

2. Aturan Gerakan Legal

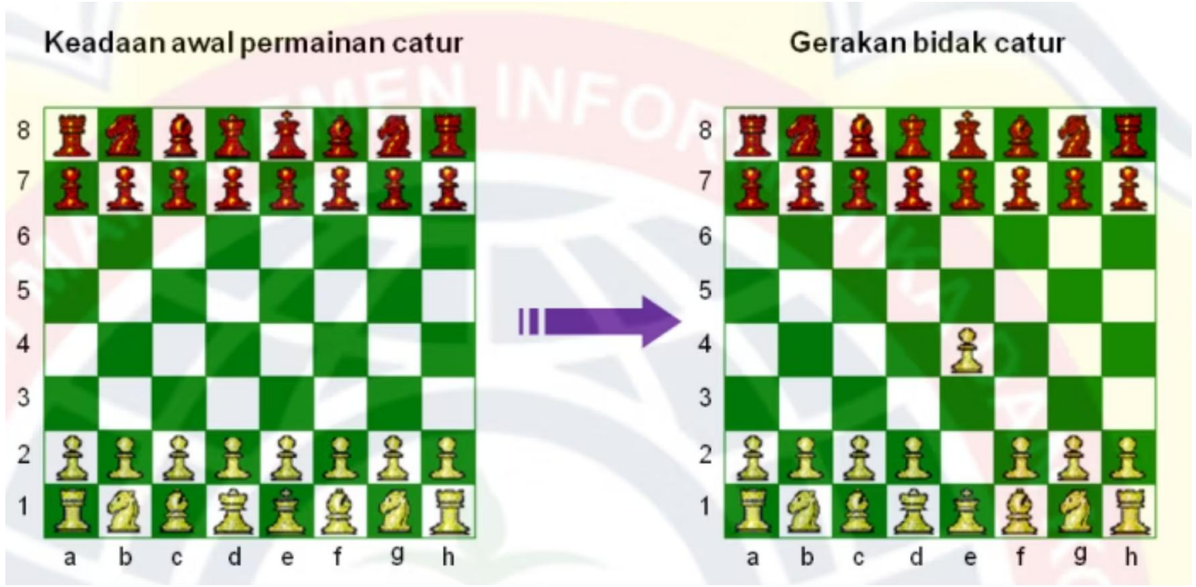
Aturan sangat berguna untuk menentukan suatu bidak bergerak dari satu keadaan ke keadaan lain. Setiap jenis bidak memiliki pola gerakan yang berbeda dan harus dipatuhi sepanjang permainan.

3. Tujuan (Goal)

Tujuan yang ingin dicapai adalah **kemenangan terhadap lawan**, yang ditunjukkan dengan posisi Raja (*King*) lawan yang tidak bisa bergerak lagi (skakmat).

Representasi Visual: Ruang Keadaan Catur

Berikut ilustrasi keadaan awal permainan catur dan contoh gerakan bidak beserta aturan (*rule*) yang mengaturnya secara formal:



Rule (aturan) :

IF bidak putih pada kotak(e,2),
and kotak(e,3) kosong,
and kotak(e,4) kosong
THEN gerakkan bidak dari (e,2) ke (e,4)

Contoh aturan dalam notasi logika: **IF** bidak putih pada kotak(e,2), **AND** kotak(e,3) kosong, **AND** kotak(e,4) kosong, **THEN** gerakkan bidak dari (e,2) ke (e,4).

Aturan-aturan ini sangat berguna untuk menentukan gerakan suatu bidak. Huruf (a,b,c,d,e,f,g,h) mewakili kolom horizontal, dan angka (1–8) mewakili baris vertikal pada papan catur.

Ruang Keadaan (*State Space*)

Definisi

Contoh catur menunjukkan representasi masalah dalam **Ruang Keadaan (State Space)**, yaitu suatu ruang yang berisi **semua keadaan yang mungkin** terjadi selama proses penyelesaian masalah berlangsung.

Cara Kerja

Kita dapat memulai bermain catur dengan menempatkan diri pada **keadaan awal**, kemudian bergerak dari satu keadaan ke keadaan yang lain sesuai dengan aturan yang ada, dan mengakhiri permainan jika salah satu pihak telah mencapai **tujuan** (goal state).

Konsep ini berlaku universal untuk semua jenis masalah yang diselesaikan dengan pendekatan AI berbasis pencarian.

Contoh Kasus: Ember Air

Kita akan mengeksplorasi **Masalah Ember Air** sebagai studi kasus representasi ruang keadaan. Masalah ini adalah contoh klasik dalam AI yang menggambarkan bagaimana aturan-aturan sederhana dapat mengarahkan sistem menuju solusi.

Studi Kasus ini Meliputi:

- Definisi ruang keadaan dengan pasangan nilai (x, y)
- Penetapan keadaan awal dan tujuan
- Formulasi aturan-aturan transisi
- Pencarian solusi melalui ruang keadaan

Deskripsi Masalah Ember Air

Perhatikan permasalahan berikut yang akan diselesaikan menggunakan pendekatan ruang keadaan:

> Ada **2 ember** masing-masing berkapasitas **4 galon (ember A)** dan **3 galon (ember B)**. Tersedia sebuah pompa dan air yang dapat digunakan untuk mengisi ember. **Bagaimana cara mengisi tepat 2 galon air ke dalam ember berkapasitas 4 galon?**

Ember A

Kapasitas: **4 galon**

Variabel: **x**

Rentang nilai: $0 \leq x \leq 4$

Ember B

Kapasitas: **3 galon**

Variabel: **y**

Rentang nilai: $0 \leq y \leq 3$

Target/Goal

Isi Ember A: **tepat 2 galon**

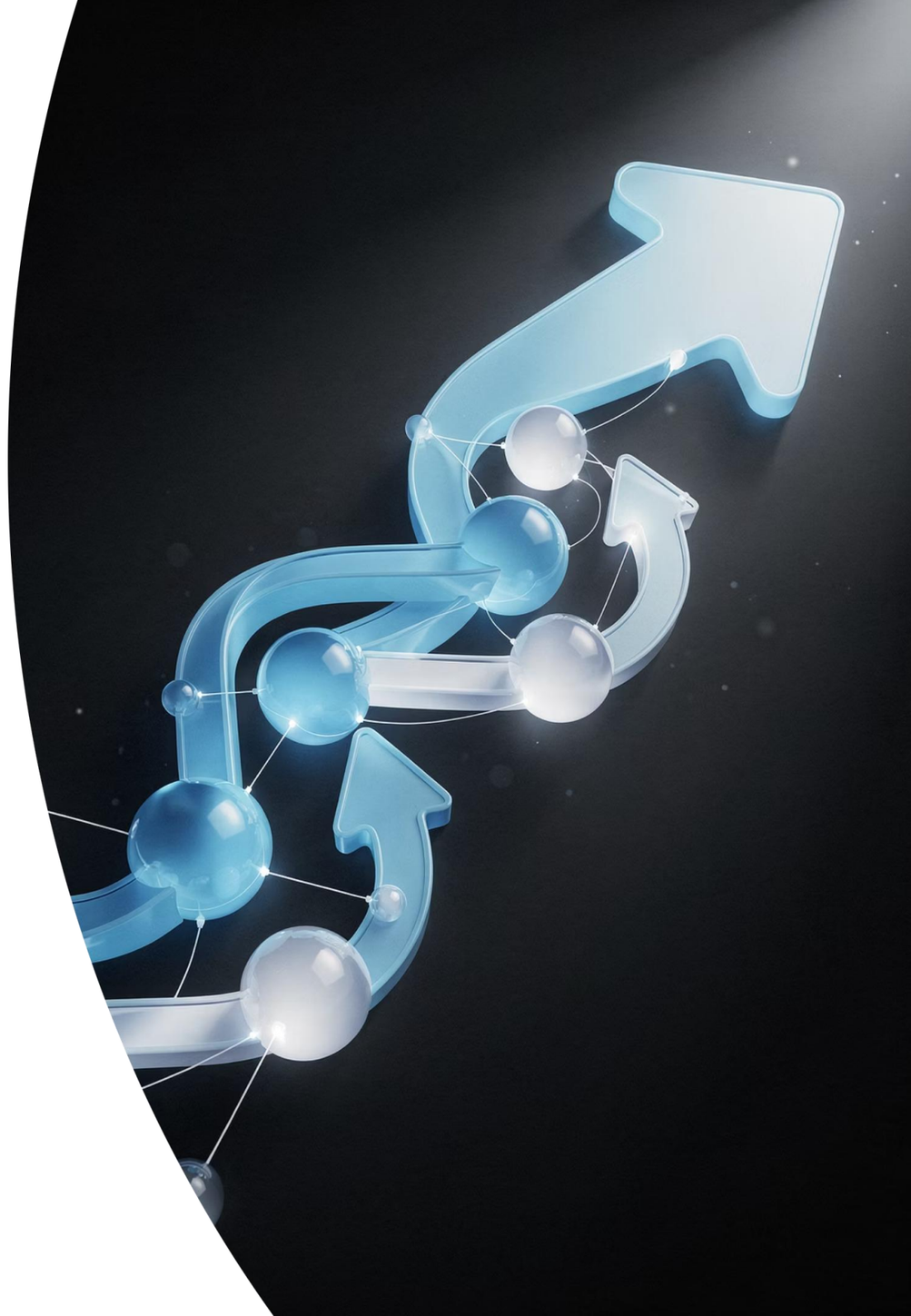
Representasi: **(2, y)**

Keadaan awal: **(0, 0)**

EMBER AIR

Penyelesaian Masalah

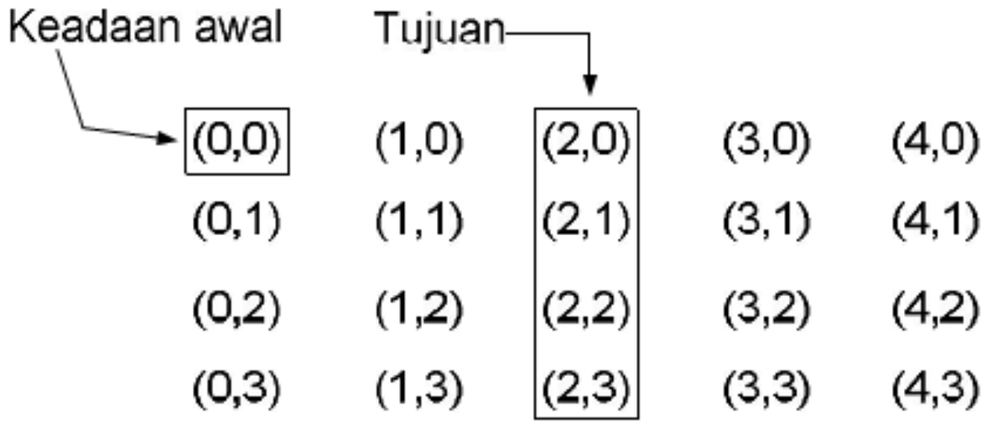
Untuk menyelesaikan masalah ember air, kita perlu mendefinisikan secara formal seluruh komponen ruang keadaan: representasi keadaan, keadaan awal, keadaan tujuan, dan aturan-aturan yang berlaku.



Ruang Keadaan Ember Air

Keadaan sistem ember air direpresentasikan sebagai pasangan nilai **(x, y)**, di mana **x** adalah isi ember A (0–4 galon) dan **y** adalah isi ember B (0–3 galon). Total ruang keadaan yang mungkin adalah **20 pasangan nilai**. Tujuan adalah mencapai keadaan **(2, y)** di mana x = 2.

Diagram di atas menggambarkan seluruh ruang keadaan. Keadaan awal ditandai sebagai **(0,0)** dan keadaan tujuan dilingkari di kolom **x=2**. Sistem harus menemukan jalur dari (0,0) menuju salah satu keadaan tujuan tersebut.



Aturan-Aturan Transisi

Diasumsikan terdapat tiga operasi dasar yang dapat dilakukan terhadap ember, yang menjadi dasar formulasi seluruh aturan transisi:

Mengisi dari Pompa

Mengisi salah satu ember dari pompa air hingga penuh (ember A hingga 4 galon, atau ember B hingga 3 galon).

Membuang Air

Membuang sebagian atau seluruh air dari salah satu ember ke luar sistem, sehingga ember menjadi kosong atau berkurang isinya.

Menuang Antar Ember

Menuangkan air dari satu ember ke ember lainnya, baik sebagian maupun seluruhnya, sesuai dengan kapasitas ember tujuan.

Dari tiga operasi dasar ini, diformulasikan **11 aturan** lengkap yang mencakup semua kemungkinan transisi keadaan yang valid.

Tabel Aturan Lengkap

Berikut adalah 11 aturan yang mendefinisikan semua transisi keadaan yang valid pada masalah ember air. Aturan-aturan ini merupakan representasi formal dari operasi yang dapat dilakukan:

No.	Kondisi (Jika)	Aksi (Maka)	Keterangan
1	(x,y), x < 4	(4,y)	Isi ember A dari pompa
2	(x,y), y < 3	(x,3)	Isi Ember B dari pompa
3	(x,y), x > 0	(x-d,y)	Tuang sebagian air dari ember A
4	(x,y), y > 0	(x,y-d)	Tuang sebagian air dari ember B
5	(x,y), x > 0	(0,y)	Kosongkan ember A
6	(x,y), y > 0	(x,0)	Kosongkan ember B
7	x+y≥4 dan y>0	(4, y–(4–x))	Tuang B ke A sampai A penuh
8	x+y≥3 dan x>0	(x–(3–y), 3)	Tuang A ke B sampai B penuh
9	x+y≤4 dan y>0	(x+y, 0)	Tuang seluruh B ke A
10	x+y≤3 dan x>0	(0, x+y)	Tuang seluruh A ke B
11	(0,2)	(2,0)	Tuang 2 galon dari B ke A

Solusi Masalah Ember Air

Terdapat **dua jalur solusi** yang berbeda untuk mencapai tujuan (2 galon di ember A). Kedua solusi ini menunjukkan bahwa ruang keadaan dapat memiliki lebih dari satu jalur menuju tujuan:

Solusi 1

Isi ember A	Isi ember B	Aturan yang dipakai
0	0	1
4	0	8
1	3	6
1	0	10
0	1	1
4	1	8
2	3	Solusi

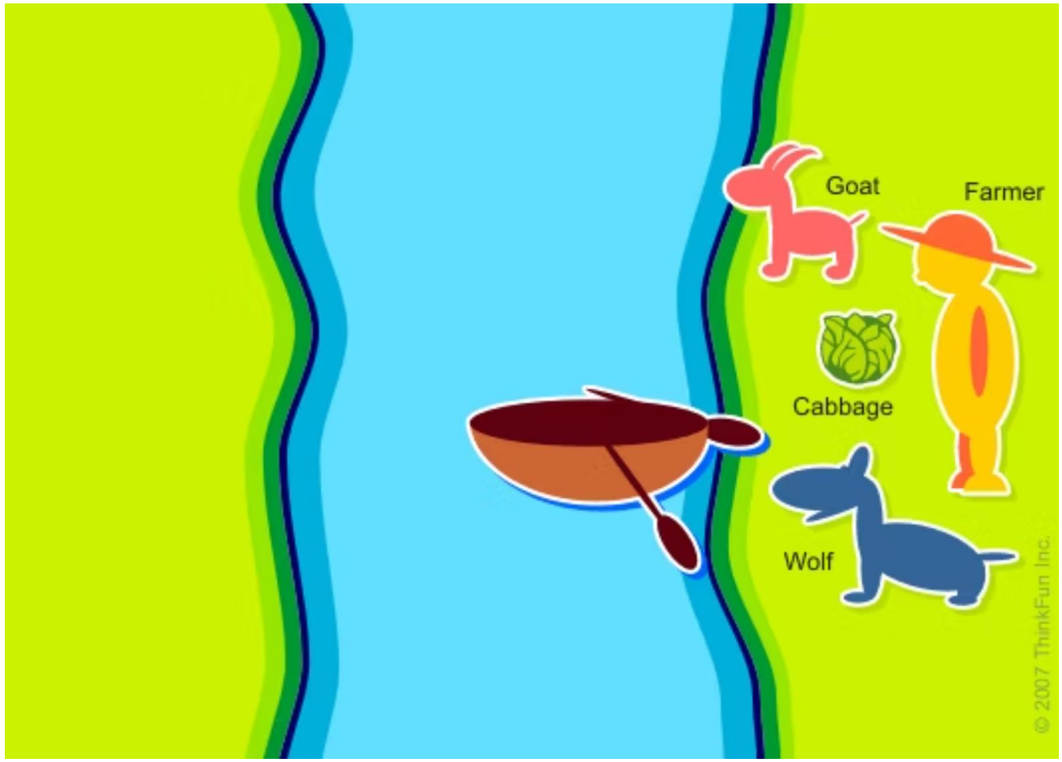
Dimulai dengan mengisi ember A, lalu menuangkan ke B, dan seterusnya hingga diperoleh 2 galon di ember A melalui 6 langkah.

Solusi 2

Isi ember A	Isi ember B	Aturan yang dipakai
0	0	2
0	3	9
3	0	2
3	3	7
4	2	5
0	2	9
2	0	Solusi

Dimulai dengan mengisi ember B terlebih dahulu, kemudian melakukan serangkaian transisi berbeda untuk mencapai solusi yang sama dalam 6 langkah.

Contoh Kasus: Puzzle Petani, Sayur, Kambing, dan Serigala



Studi kasus kedua adalah **Puzzle Petani** — salah satu masalah klasik dalam literatur AI dan logika komputasional. Masalah ini menguji kemampuan sistem dalam menavigasi ruang keadaan dengan **batasan (constraints)** yang kompleks.

Deskripsi Masalah: Puzzle Petani

Berikut adalah deskripsi lengkap permasalahan yang harus diselesaikan:

Situasi Awal

Seorang petani akan menyeberangkan seekor **kambing**, seekor **serigala**, dan **sayur-sayuran** melewati sungai menggunakan sebuah perahu (boat).

Batasan Kapasitas

Perahu hanya bisa memuat **petani dan satu penumpang lain** saja, baik kambing, serigala, maupun sayur-sayuran dalam setiap perjalanan.

Batasan Keamanan

Jika ditinggalkan oleh petani: **sayur-sayuran akan dimakan oleh kambing**, dan **kambing akan dimakan oleh serigala**. Kombinasi berbahaya ini harus selalu dihindari.

Tujuan: Seberangkan semua (kambing, serigala, sayuran, dan petani) ke daerah tujuan tanpa ada yang dimakan.

Langkah Penyelesaian Masalah

Metodologi penyelesaian masalah dalam AI berbasis ruang keadaan selalu mengikuti empat langkah fundamental berikut. Urutan ini berlaku untuk semua jenis masalah pencarian:

1

Definisikan Ruang Keadaan

Tentukan representasi formal semua kemungkinan konfigurasi sistem.

2

Tetapkan Keadaan Awal

Tentukan satu atau lebih kondisi awal sistem sebelum proses dimulai.

3

Tetapkan Tujuan

Tentukan satu atau lebih kondisi akhir yang ingin dicapai oleh sistem.

4

Tetapkan Aturan

Formulasikan kumpulan aturan yang mengatur transisi antar keadaan secara valid.

Identifikasi Ruang Keadaan

Permasalahan puzzle petani dapat dilambangkan dengan notasi **(JumlahKambing, JumlahSerigala, JumlahSayuran, JumlahBoat)** pada daerah asal. Setiap variabel bernilai **0 atau 1**, mewakili kehadiran entitas di daerah asal.

Nilai 1

Entitas **ada** di daerah asal (belum menyeberang ke daerah tujuan).

Nilai 0

Entitas **tidak ada** di daerah asal (telah menyeberang ke daerah tujuan).

Contoh: (0,1,1,1)

Pada daerah asal: **tidak ada kambing**, ada serigala, ada sayuran, dan ada boat — kambing sudah menyeberang.

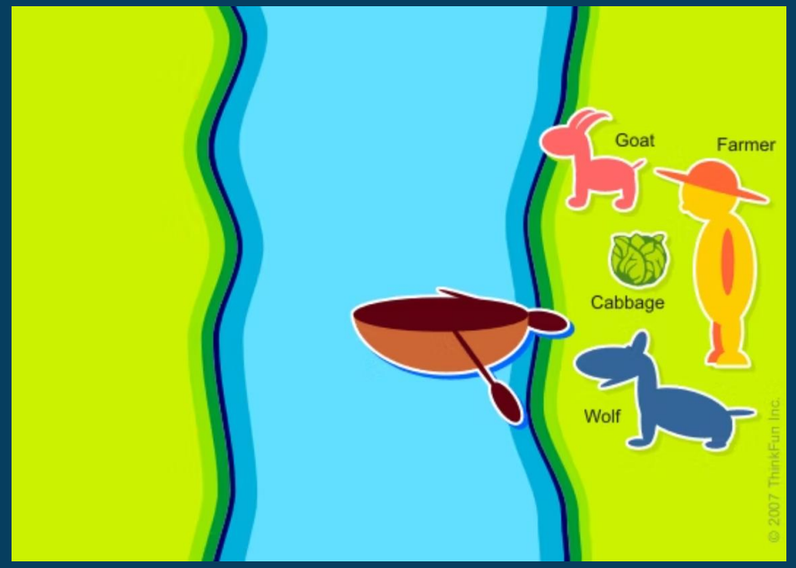
Total ruang keadaan yang mungkin adalah 2^4 = **16 kombinasi**, namun tidak semua valid karena adanya batasan keamanan.

Keadaan Awal dan Tujuan

Keadaan Awal

Daerah Asal: (1,1,1,1) Kambing, serigala, sayuran, dan boat semuanya ada di daerah asal.

Daerah Seberang: (0,0,0,0) Belum ada satu pun entitas yang telah menyeberang.



Keadaan Tujuan (Goal State)

Daerah Asal: (0,0,0,0) Tidak ada lagi entitas yang tersisa di daerah asal.

Daerah Seberang: (1,1,1,1) Semua entitas — kambing, serigala, sayuran, dan boat — telah berhasil menyeberang dengan selamat.

Sistem harus menemukan urutan perpindahan yang **tidak pernah meninggalkan kombinasi berbahaya** tanpa pengawasan petani.

Aturan-Aturan Transisi

Terdapat **7 aturan** yang mendefinisikan semua perpindahan valid dalam puzzle petani. Setiap aturan menggambarkan satu aksi yang dapat dilakukan petani:

Aturan Ke-	Aturan	Arah
1	Kambing menyeberang (bersama petani)	Asal → Tujuan
2	Sayuran menyeberang (bersama petani)	Asal → Tujuan
3	Serigala menyeberang (bersama petani)	Asal → Tujuan
4	Kambing kembali (bersama petani)	Tujuan → Asal
5	Sayuran kembali (bersama petani)	Tujuan → Asal
6	Serigala kembali (bersama petani)	Tujuan → Asal
7	Boat kembali (petani tanpa penumpang)	Tujuan → Asal

Setiap penerapan aturan harus memenuhi **batasan keamanan**: kambing tidak boleh bersama sayuran tanpa petani, dan serigala tidak boleh bersama kambing tanpa petani.

Solusi Lengkap: Puzzle Petani

Berikut adalah urutan langkah solusi yang berhasil memindahkan semua entitas dari daerah asal ke daerah tujuan tanpa pelanggaran batasan keamanan:

Daerah Asal	Daerah Tujuan	Aturan	Aksi
(1,1,1,1)	(0,0,0,0)	1	Kambing menyeberang
(0,1,1,0)	(1,0,0,1)	7	Boat kembali
(0,1,1,1)	(1,0,0,0)	3	Serigala menyeberang
(0,0,1,0)	(1,1,0,1)	4	Kambing kembali
(1,0,1,1)	(0,1,0,0)	2	Sayuran menyeberang
(1,0,0,0)	(0,1,1,1)	7	Boat kembali
(1,0,0,1)	(0,1,1,0)	1	Kambing menyeberang
(0,0,0,0)	**(1,1,1,1)**	—	**SOLUSI ✓**

Notasi: **(JumlahKambing, JumlahSerigala, JumlahSayuran, JumlahBoat)** di daerah asal.

Ada Pertanyaan?

Silakan ajukan pertanyaan seputar materi **Masalah dan Ruang Keadaan** yang telah dibahas. Diskusi aktif sangat dianjurkan untuk memperdalam pemahaman konsep!

Ruang Keadaan

Bagaimana cara mendefinisikan state space untuk masalah baru?

Aturan Transisi

Bagaimana memastikan aturan yang dibuat lengkap dan tidak bertentangan?

Pencarian Solusi

Strategi apa yang paling efisien untuk menavigasi ruang keadaan yang besar?

Latihan: Puzzle Pendekar-Monster

Definisikan secara lengkap komponen ruang keadaan, keadaan awal, tujuan, aturan-aturan, hingga penyelesaian untuk masalah berikut:

Deskripsi Masalah

Ada **3 Pendekar** dan **3 Monster** yang hendak menyeberang sungai. Hanya tersedia **1 perahu** yang dapat membawa maksimal **2 orang** sekaligus. Diasumsikan perahu dapat kembali sendiri.

Batasan Kritis

Jika pada suatu lokasi (daerah asal atau daerah tujuan) jumlah **monster lebih banyak dari pendekar**, maka monster akan memakan pendekar. Kondisi ini harus selalu dihindari di kedua sisi sungai.

Yang Harus Dikerjakan

- Identifikasi ruang keadaan dengan notasi yang tepat
- Tetapkan keadaan awal dan keadaan tujuan
- Formulasikan aturan-aturan transisi yang valid
- Temukan urutan langkah solusi lengkap