

Alpha Beta Pruning

Kelompok 8

Anggota Kelompok :

Abdurrahman Farimza	5025201125
Kartika Diva Asmara Gita	5025211039
Hana Maheswari	5025211182

Adversarial Search



Adversarial search adalah sebuah pendekatan dalam kecerdasan buatan yang digunakan untuk memecahkan masalah di mana ada dua atau lebih agen yang saling bersaing. Pendekatan ini didasarkan pada konsep 'Game Theory' yang mana untuk menyelesaikan permainan harus dimenangkan oleh satu pihak dan pihak lainnya kalah secara otomatis. Ada 2 algoritma yang digunakan yaitu, minimax atau alpha beta pruning.



Perbedaan Minimax dan Alpha Beta Pruning

1. Pencarian Simpul:

- Minimax: semua simpul dalam ruang keadaan permainan dieksplorasi secara penuh untuk menentukan langkah terbaik.
- Alpha-Beta Pruning: menghapus beberapa simpul dari pencarian yang tidak akan mempengaruhi hasil akhir dengan memanfaatkan informasi dari simpul yang telah dievaluasi sebelumnya.

2. Kompleksitas Waktu:

- Minimax: $O(b^d)$, di mana b adalah faktor cabang pergerakan rata-rata dan d adalah kedalaman pencarian.
- Alpha-Beta Pruning: secara umum $O(b^{(d/2)})$ karena algoritma ini memangkas sebagian besar pencarian yang tidak relevan.

3. Keuntungan dan Kekurangan:

- Minimax: menjamin solusi optimal jika semua simpul dieksplorasi, tetapi kompleksitas waktu yang tinggi membuatnya tidak efisien dalam permainan yang kompleks.
- Alpha-Beta Pruning: tidak menjamin solusi optimal, tetapi dapat memberikan solusi yang baik dengan mengurangi jumlah simpul yang diperiksa sehingga lebih efisien.

Algoritma

1. Inisialisasi:

Terdapat parameter alpha dan beta yang digunakan untuk melacak batasan atas dan bawah untuk memotong cabang-cabang pencarian yang tidak relevan. Alpha adalah batas bawah untuk nilai terbaik yang telah ditemukan oleh pemain saat ini. Beta adalah batas atas untuk nilai terbaik yang telah ditemukan oleh lawan.

- Tetapkan nilai alpha ke minus tak hingga (negative infinity).
- Tetapkan nilai beta ke plus tak hingga (positive infinity).

2. Pencarian Minimax:

- Mulai pencarian minimax pada simpul akar.
- Evaluasi simpul saat ini berdasarkan status permainan dan langkah-langkah yang telah dilakukan.
- Jika simpul saat ini adalah simpul terminal (misalnya, permainan berakhir atau mencapai batas kedalaman), kembalikan nilai keuntungan (misalnya, skor) dari simpul tersebut.

Algoritma

3. Rekursi:

a. Jika simpul saat ini adalah simpul pemain saat ini, lakukan langkah-langkah berikut:

- Untuk setiap langkah yang mungkin, lakukan pencarian minimax pada simpul anak.
- Perbarui nilai alpha dengan maksimum antara nilai alpha saat ini dan nilai hasil pencarian minimax dari simpul anak.
- Jika nilai alpha saat ini lebih besar dari atau sama dengan beta, hentikan pencarian di simpul saat ini dan lakukan pruning. Kembalikan nilai alpha sebagai hasil pencarian.

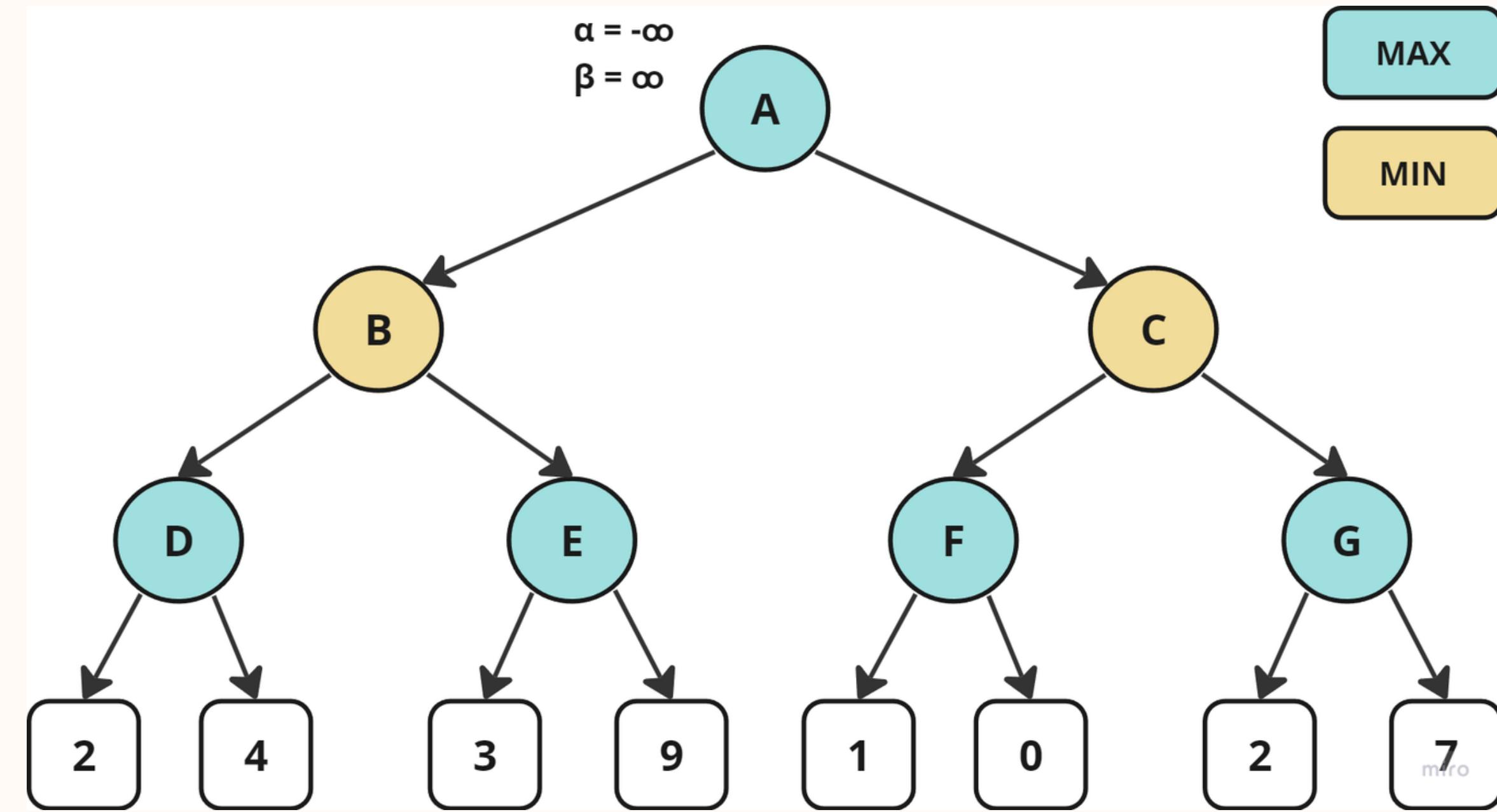
b. Jika simpul saat ini adalah simpul lawan, lakukan langkah-langkah berikut:

- Untuk setiap langkah yang mungkin, lakukan pencarian minimax pada simpul anak.
- Perbarui nilai beta dengan minimum antara nilai beta saat ini dan nilai hasil pencarian minimax dari simpul anak.
- Jika nilai beta saat ini kurang dari atau sama dengan alpha, hentikan pencarian di simpul saat ini dan lakukan pruning. Kembalikan nilai beta sebagai hasil pencarian.

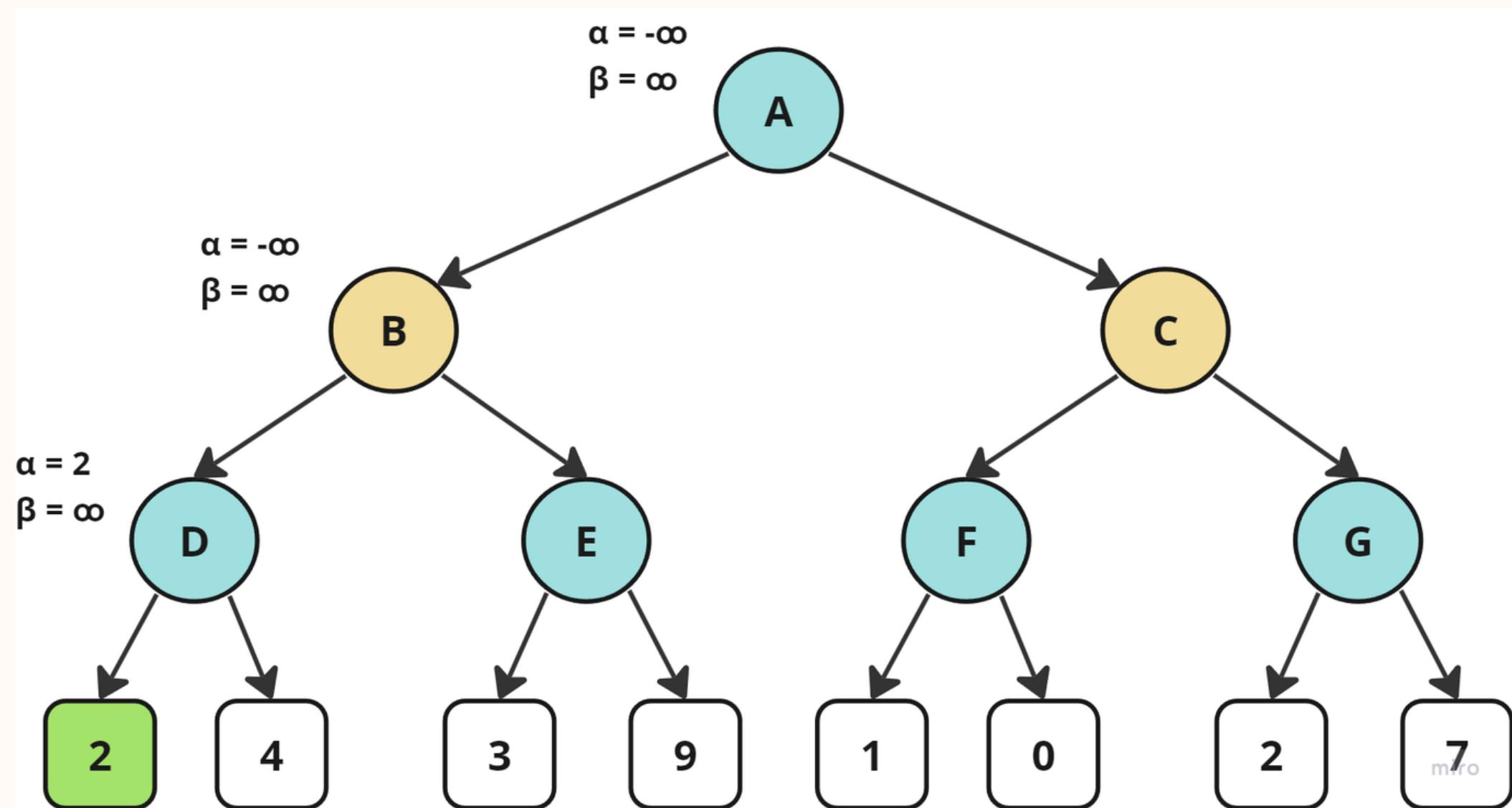
Algoritma

4. Kembali ke Akar:

- Setelah pencarian selesai, kembali ke simpul akar.
- Pilih langkah dengan nilai hasil pencarian tertinggi jika pemain saat ini adalah pemain maksimum, atau pilih langkah dengan nilai hasil pencarian terendah jika pemain saat ini adalah pemain minimum.



- Root node adalah maximizer karena mewakili giliran pemain yang memaksimalkan. Tujuan pemaksimalan adalah untuk memilih nilai maksimum di antara node anaknya.
- Level berikutnya berisi node minimizer karena mewakili giliran lawan. Tujuan minimizer adalah untuk memilih nilai minimum di antara node anaknya.
- Level kedua terakhir berisi simpul pemaksimal lagi, yang mewakili giliran pemain pemaksimal.
- Level terakhir mewakili node terminal, yang memiliki nilai 2, 4, 3, 9, 1, 0, 2, dan 7. Ini adalah hasil akhir atau skor permainan untuk berbagai kemungkinan gerakan.



Di simpul akar (A):

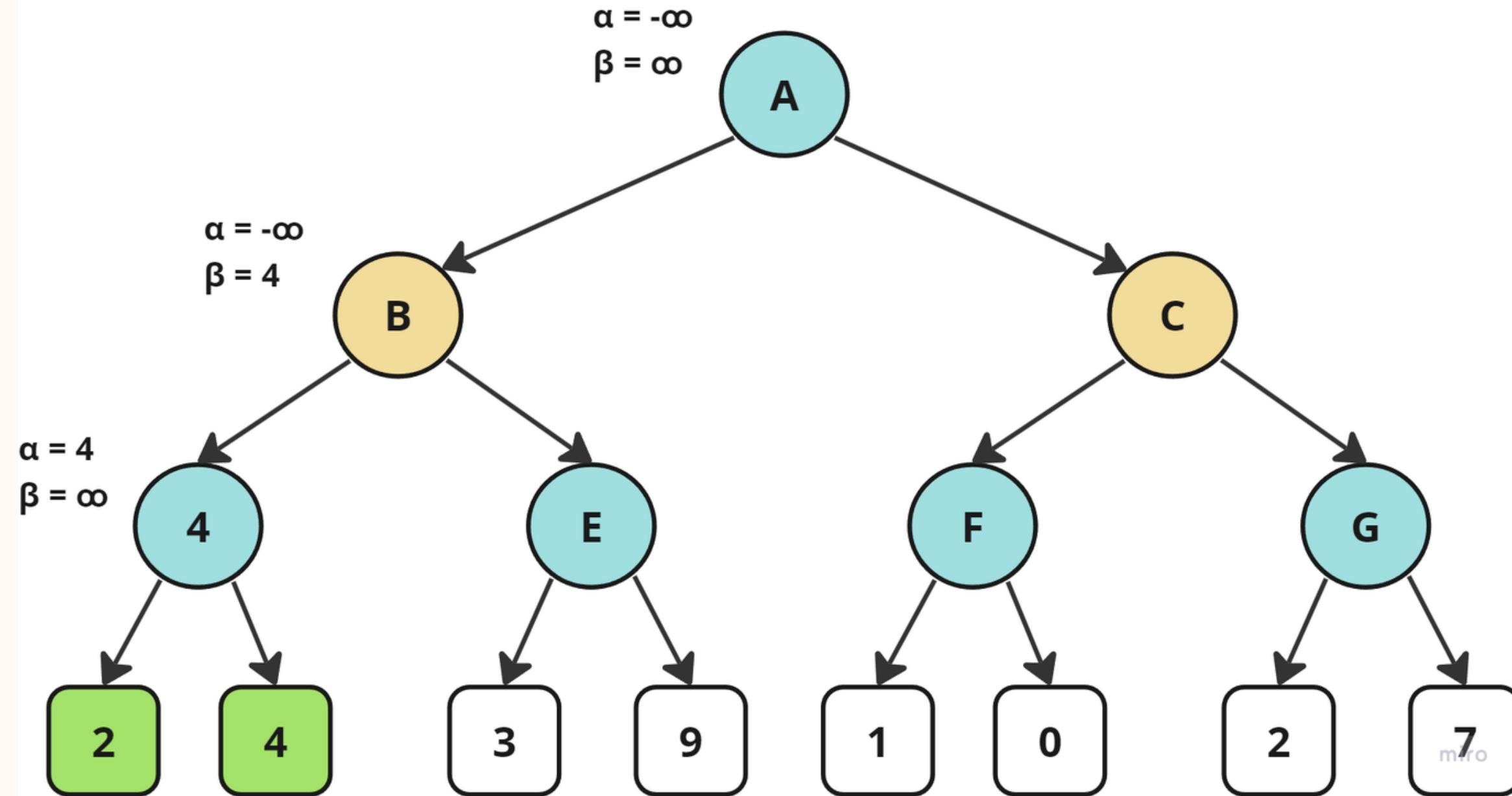
- Nilai awal alpha adalah $-\infty$ (negatif infinity).
- Nilai awal beta adalah $+\infty$ (positif tak terhingga).

Pada simpul peminimal B:

- Nilai alpha dari B tetap $-\infty$.
- Nilai beta dari B tetap $+\infty$.

Pada simpul pemaksimal D:

- Nilai alpha D diperbarui menjadi 2 karena nilai node terminal paling kiri (2) lebih besar dari nilai alpha awal ($-\infty$).

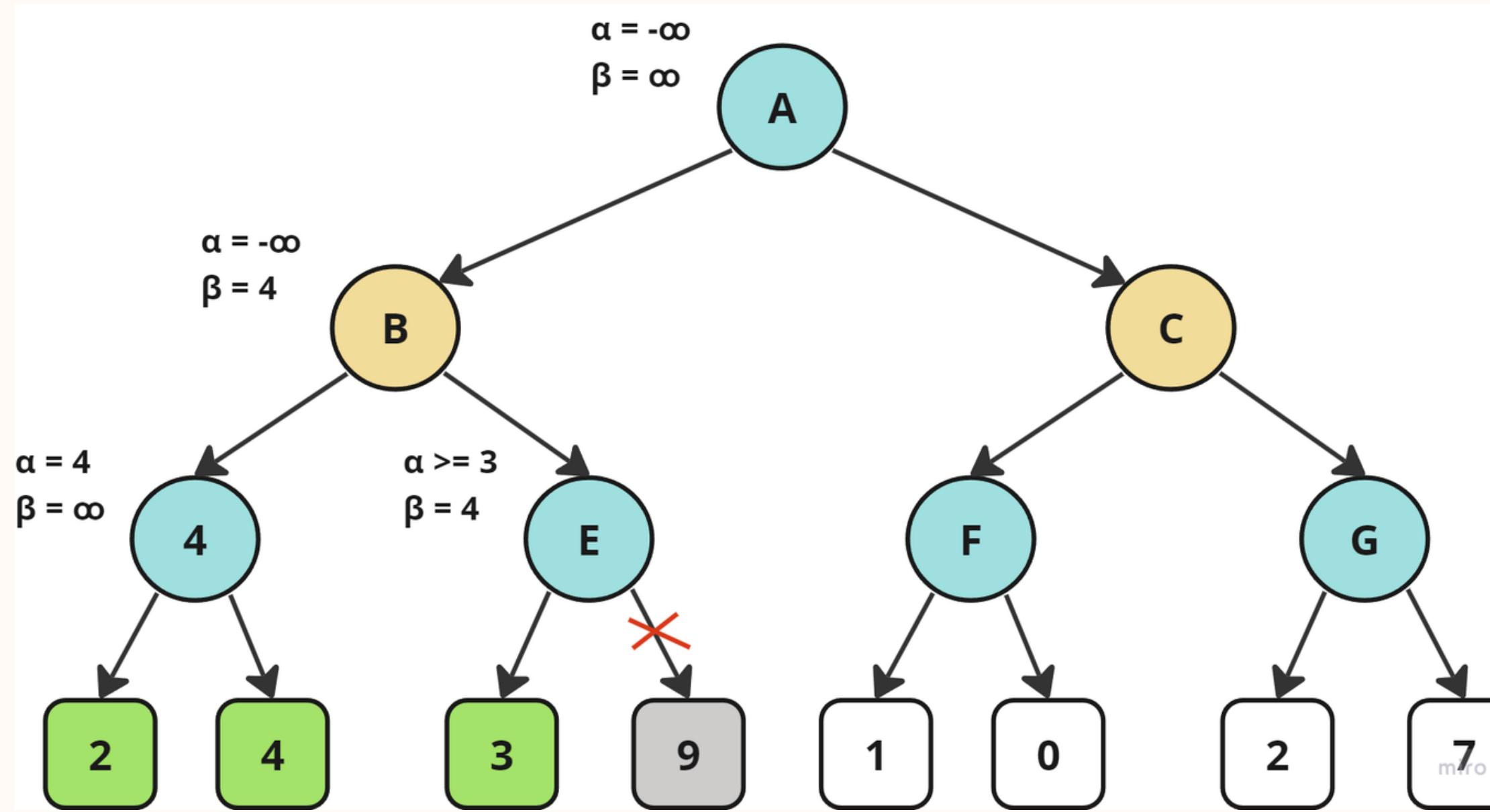


- Pada simpul pemaksimal D:

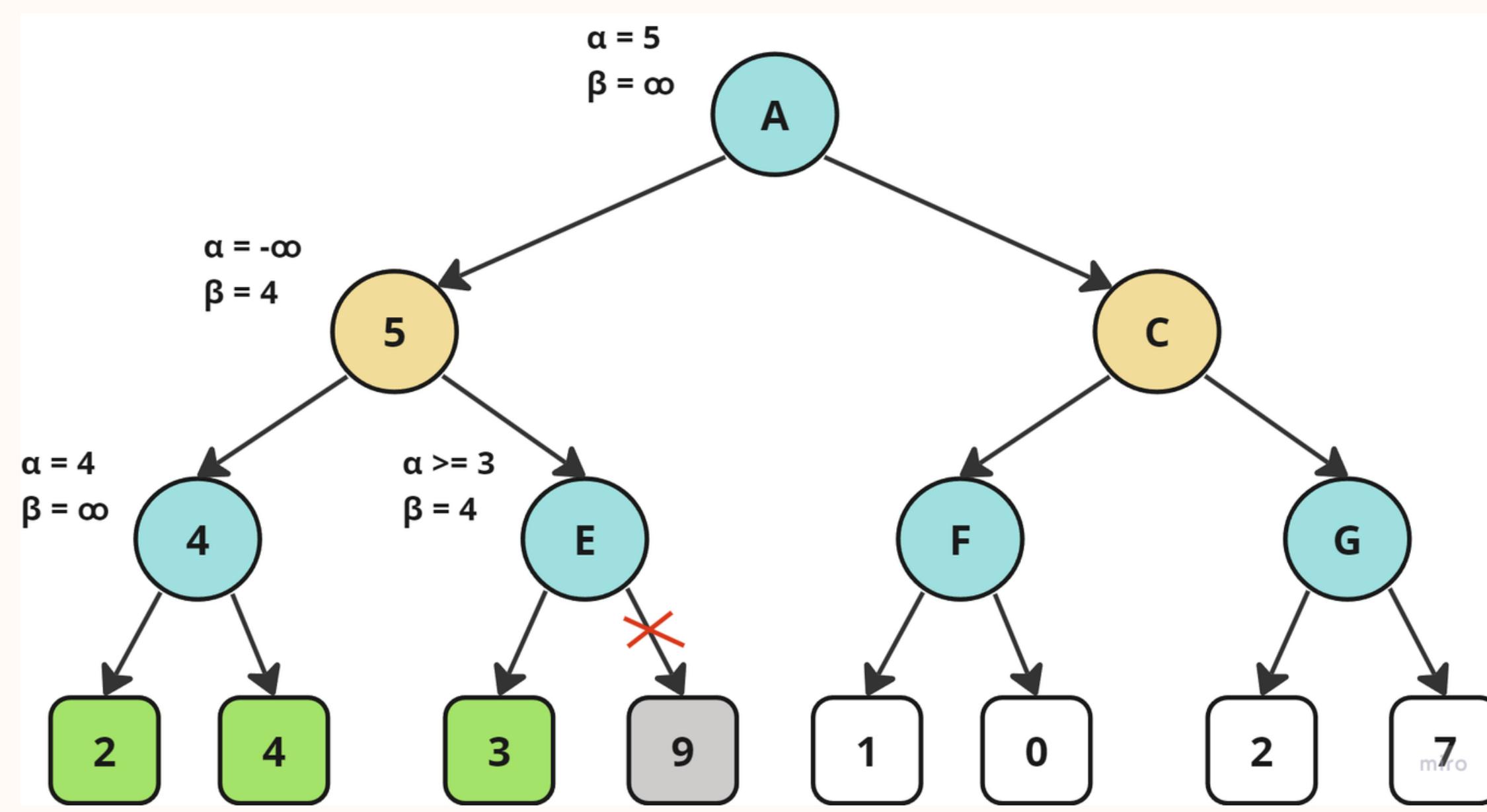
Setelah mengevaluasi anak terminal kanan D, yang memiliki nilai 4, dilakukan perbandingan dengan nilai alfa saat ini (2) dan memperbarui nilai alfa D menjadi 4 karena 4 lebih besar dari 2.

- Pindah ke atas ke simpul minimizer B

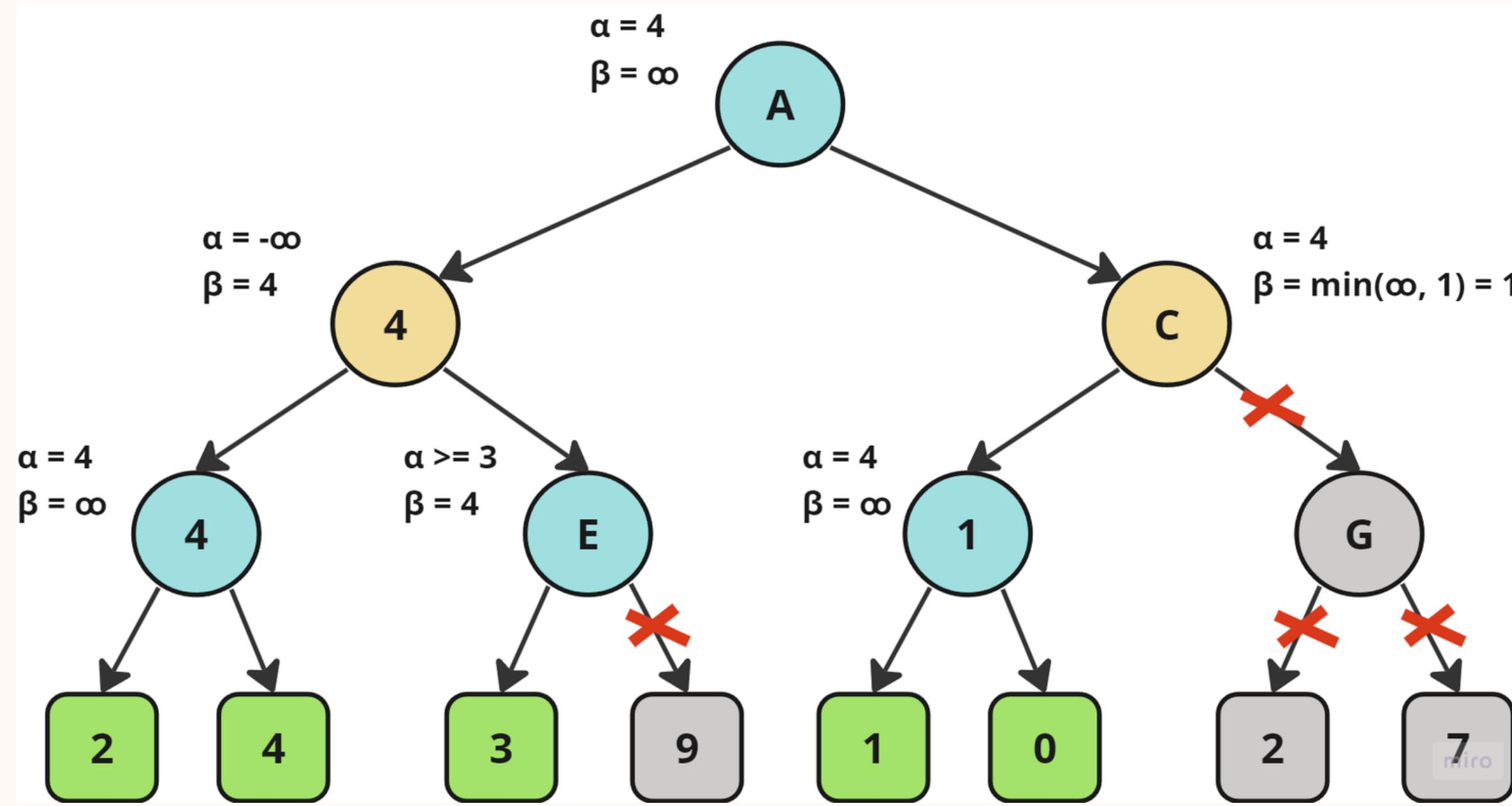
Perbarui nilai beta B menjadi 4 karena nilai D (anak B) kurang dari atau sama dengan 4 sehingga 4 adalah batas atas baru untuk B.



- Pada simpul minimizer E, nilai node terminal kiri adalah 3 sehingga perbarui nilai alfa E menjadi 3 karena nilai simpul terminal kiri lebih besar dari nilai alfa saat ini ($-\infty$).
- Karena B adalah node minimizer, perlu memeriksa apakah ada kemungkinan mendapatkan nilai yang lebih kecil dari node anaknya. Karena nilai anak kanan E (9) melebihi nilai alfa sehingga tidak perlu menjelajahi anak kanan E dan tidak memperbarui nilai beta E karena nilai yang diinginkan seharusnya lebih kecil.



- Perbarui nilai B dari pencarian kedua anak B dan ubah nilai alfa pada simpul akar.
- Pindah ke subtree kanan untuk mengevaluasi anak pertama dari F. Perbarui nilai alfa F menjadi 1 karena nilai simpul terminal kiri lebih besar dari nilai alfa saat ini ($-\infty$). Ini menandakan bahwa nilai F akan lebih besar atau sama dengan 1.
- Lakukan perbandingan nilai F (1) dengan nilai beta saat ini ($+\infty$) dan memperbarui nilai beta C menjadi 1. Hal ini menunjukkan bahwa nilai C akan lebih kecil atau sama dengan 1.



- Disimpulkan, nilai C akan kurang dari atau sama dengan 1. Tetapi pada simpul akar memerlukan nilai lebih besar dari atau sama dengan 4, yang tidak mungkin dilakukan oleh C sehingga perlu memangkas sisanya dari anak-anak C.
- Setelah menjelajahi subtree kiri dan kanan dari simpul akar, dapat disimpulkan bahwa nilai terbaik dari A adalah 4 dan yang dapat dicapai dari jalur yang diberikan A \rightarrow B \rightarrow D \rightarrow 4.



TERIMA KASIH