

ІТМО

Введение в GameDev: ОСНОВЫ ИГРОВОГО ИИ

Лекция 1. Введение в курс. Теория игр. Поиск по дереву.



1. Организационная часть
2. Введение в игровой ИИ
3. Элементы теории игр
4. Поиск по дереву



Атыпшева Татьяна Владимировна



- ☐ Магистрант ИТМО по программе “Мультимедиа-технологии, дизайн и юзабилити”
- ☐ Окончила больше 15 коммерческих курсов в области GameDev
- ☐ Инженер в международной GameDev компании в СПб
- ☐ Призер международных GameDev хакатонов
- ☐ Исследователь в области GameDev (влияние аудиовизуальных составляющих в хоррор играх, методы игрового ИИ для вариативного поведения, обучающие игровые приложения)



Абросимов Кирилл Игоревич

- ☐ Окончил магистратуру ИТМО по программе “Речевые технологии и машинное обучение”
- ☐ Практикующий Data Scientist: ML, DL, MIR, OR, NLP
- ☐ Аспирант Университета Виктории (CS, major: MIR)
- ☐ Исследователь в области интеллектуальных музыкальных и речевых технологиях

ГЛАВА 1 _ Присоединяйтесь



<https://t.me/+4mQ12iZbtmxxhMGly>



Сейчас я разделю вас на группы, где вы познакомитесь:

1. Расскажите о себе: где учитесь, почему решили присоединиться к факультативу
2. Расскажите о своих навыках в гейм деве, возможно, проектах

“Расставим точки над і” или “Ожидание-Реальность”

Что вы ожидаете от курса?

Что бы вы **хотели** видеть на курсе

PS. Пишем в чат!



“Расставим точки над і” или “Ожидание-Реальность”

Что вы ожидаете от курса?

Что бы вы **не хотели** видеть на курсе

PS. Пишем в чат!

“Расставим точки над i” или “Ожидание-Реальность”



1. Что такое ОУ факультатив?

Факультатив - это **образовательно-развлекательное** мероприятие в ИТМО для получения новых навыков и знаний для любого студента ИТМО

(с) Татьяна Атяпшева

2. У меня изменилось расписание, факультатив не оправдал ожиданий, нашел новую работу: как я могу отписаться?

Напишите на почту ОУФ plus@itmo.ru и предупредите преподавателя.

Больше информации: https://student.itmo.ru/ru/university_electives/

3. Какой Unreal Engine мы будем использовать на практиках?

Используется исключительно Unreal Engine 4 (4.26). Обуславливается тем, что на факультатив могут присоединиться студенты не с мощными ЭВМ (**кейсы были**). А основы ИИИ полученные на UE4 вы сможете применить в будущем и в UE5.

“Расставим точки над i” или “Ожидание-Реальность”



4. **А что если я буду делать все на UE5?**

Преподаватель не будет проверять. В карточке курса версия UE была указана, а значит, вы приняли заданные правила игры, когда записались на факультатив.

5. **Относительно игровиков: насколько глубоко будем лезть в ИИИ в UE?**

Не глубоко. Вряд ли вы получите больше навыков разработки ИИИ в UE, чем на курсе “Основы разработки компьютерных игр”. А далее у вас там годовой курс ИИИ, годовой курс интеллектуальной генерации контента. Но и сравнивать **major** с **факультативом** - **не имеет смысла**.

6. **Относительно игровиков: так а нам стоит оставаться вообще?**

Да, если хотите изучить теоретические основы ИИИ, еще раз попрактиковаться в разработке ИИИ, создать итоговый проект. А также показать себя в треке “Тру прогеры” в рамках практических работ.

I. **Blueprints и GIT**















Вы можете разрабатывать простые сценарии с помощью blueprints: Спавн и уничтожение чего-либо, работа с переменными и материалами в игре, логика начала и конца игры, работа с основными нодами и функциями (флоу контроль и проч.). Вы работали с системами контроля версий для командной разработки чего-либо.

II. **Основы высшей и прикладной математики**

Вы знаете основы дискретной математики, основные структуры данных, в общих чертах понимаете какие оптимизационные задачи бывают и как их решать (исследование операций / методы оптимизации / математическое программирование).

ГЛАВА 1 _ Структура курса

Лекции		Практики (UE4)	
29.10.24	 Теория игр и поиск по дереву	02.11.24	 Введение в ИИИ
05.11.24	 Ad-Hoc Behavior Authoring	09.11.24	 AI контроллер и навигация
12.11.24	 Технологии программирования ИИИ	16.11.24	 Деревья поведений I
19.11.24	 Принятие решений и элементы машинного обучения	23.11.24	 Деревья поведений II
26.11.24	 Нечеткие модели и системы	30.11.24	 Технологии программирования ИИИ
03.12.24	 Эволюционные и роевые алгоритмы	07.12.24	 Защита проектов
14.12.24	Зачет		

ГЛАВА 1 _ Структура курса

Оценочное средство	Балл	Точка
ЛР1. Навигация в UE4	10	
ЛР2. Деревья поведений в UE4	10	
Проект. Применение методов игрового ИИ в UE4	30	10+
ПР1. Поиск по дереву	8	
ПР2. Принятие решений	7	
ПР3. Нечеткие модели и системы	8	
ПР4. Эволюционные и роевые алгоритмы	7	
Зачет. Рабочая тетрадь.	20	10+



Зачет:

1. 60+ баллов
2. Проект 10+
3. Тетрадь 10+

ГЛАВА 1 _ Рабочая тетрадь

ИТМО

ИТМО

Рабочая тетрадь

ПО КУРСУ

Введение в GameDev:
Основы игрового ИИ



Студент:

Группа:

Семестр: Осень Год:



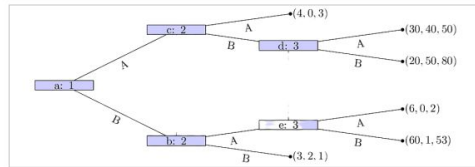
ЛЕКЦИЯ 1. ВВЕДЕНИЕ В ТЕОРИЮ ИГР И ПОИСК ПО ДЕРЕВУ.

\ Элементы теории игр /

В теории игр рассматриваются игры в двух формах. Игры в форме описываются **матрицей выигрыша** (платежной матрицей). Например, для игры «Числа»: «Два человека независимо друг от друга записывают одно число 1, 2 или 3. Выигрыш равен сумме чисел, причем, если сумма четное число, то выигрывает первый игрок, если нечетное, то второй», матрица выигрыша будет выглядеть следующим образом:

И1 \ И2	1	2	3
1	<input type="text"/>	<input type="text"/>	<input type="text"/>
2	<input type="text"/>	<input type="text"/>	<input type="text"/>
3	<input type="text"/>	<input type="text"/>	<input type="text"/>

Игры в форме характеризуются (шахматы, шашки, ГО и т.д.). (англ. Backward induction) - это итеративный процесс рассуждения в обратном направлении от конечных ситуаций в к актуальному (или стартовому) состоянию игры для решения игр в форме и вывода последовательности оптимальных действий. Он используется в теории игр для определения наиболее оптимальной последовательности действий, основанной на рациональном поведении.



2 | ИТМО

Альфа отсечение. Правило:

Бета отсечение. Правило:



Альфа-бета процедура всегда приводит к тому же результату () , что и простая минимаксная процедура той же глубины.

Score: /4

Литература:

1. Georgios N. Yannakakis and Julian Togelius. «[Artificial Intelligence and Games](#)». Springer, 2018
2. Мазалов В. В. «[Математическая теория игр и приложения: Учебное пособие](#)». — 2е изд., стер. — СПб.: Издательство «Лань», 2016. — 448 с.: ил.
3. Stuart Russell and Peter Norvig. «[Artificial Intelligence: A Modern Approach](#)». Prentice-Hall, Englewood Cliffs, 1995 (3d edition 2010)
4. Эндрю А. «[Искусственный интеллект](#)» \Пер. с англ./Под ред. и с предисл. Д. А. Поспелова - Москва: Мир, 1985 - с.264
5. Е.И. Большакова, М.Г. Мальковский, В.Н. Пильщиков. «[Искусственный интеллект. Алгоритмы эвристического поиска](#)», М.: Издательский отдел факультета ВМК МГУ - 2002, с.83

6 | ИТМО

ГЛАВА 1 _ Литература

Основной источник

Georgios N. Yannakakis and Julian Togelius

Artificial Intelligence and Games

January 26, 2018

Хранится тут

← Введение в GameDev: основы игрового ИИ



Лекции

Литература

Оценочные
средства

Семинары

В лекциях сноска

Литература в РТ

Литература:

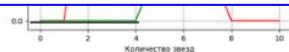
1. Georgios N. Yannakakis and Julian Togelius. «[Artificial Intelligence and Games](#)». Springer, 2018
2. Мазалов В. В. «[Математическая теория игр и приложения: Учебное пособие](#)». — 2е изд., стер. — СПб.: Издательство «Лань», 2016. — 448 с.: ил.
3. Stuart Russell and Peter Norvig. «[Artificial Intelligence: A Modern Approach](#)». Prentice-Hall, Englewood Cliffs, 1995 (3d edition 2010)
4. Эндрю А. «[Искусственный интеллект](#)» \Пер. с англ./Под ред. и с предисл. Д. А. Поспелова - Москва: Мир, 1985 - с.264
5. Е.И. Большакова, М.Г. Мальковский, В.Н. Пильщиков. «[Искусственный интеллект. Алгоритмы эвристического поиска](#)», М.: Издательский отдел факультета ВМК МГУ - 2002, с.83

6 | ИТМО

*Черноиванова Е.А. Курс лекций "Исследование операций и методы оптимизации"

Об Исследовании операций от System Lab

НОВ ИНТУИТ | Эволюционные вычисления | Информация (intuit.ru)



Борисов В. В., Круглов В. В., Федулов А. С. "Нечеткие модели и сети." — 2-е изд., стереотип. — М.:Горячая линия—Телеком, 2018 — 284 с.: ил.

Min-Sum model

AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors By Alex J. Champandard, 2003

Презентация по FRBS

Дорофеев, Максим

Джедайские техники. Как воспитать свою обезьяну, опустошить инбокс и сберечь мыслотопливо / Максим Дорофеев. — 3-е изд., испр. и доп. — М.: Манн, Иванов и Фербер, 2018. — 368 с.

ISBN 978-5-00117-065-5

Применение на практике



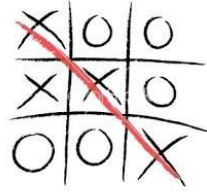
Полезная статья про прикладную навигацию в играх

ГЛАВА 1 _ Игровой ИИ: intro

ИТМО



ГЛАВА 1 _ Игровой ИИ: немного истории



Minimax algo

OXO, Tic Tac Toe

Rote learning

Alpha-Beta pruning

TD-Gammon
(RL + NN)

Alan Turing
1946

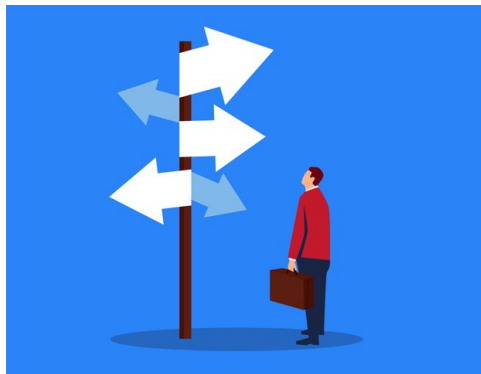
Alexander Douglas
1952

Arthur Samuel
1950s

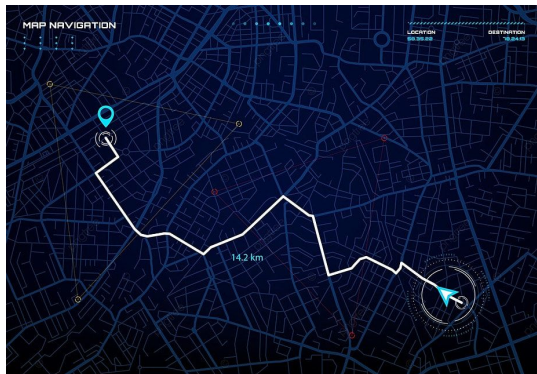
Gerald Tesauro
1992



ГЛАВА 1 _ Игровой ИИ: задачи



Принятие решений



Движение и навигация

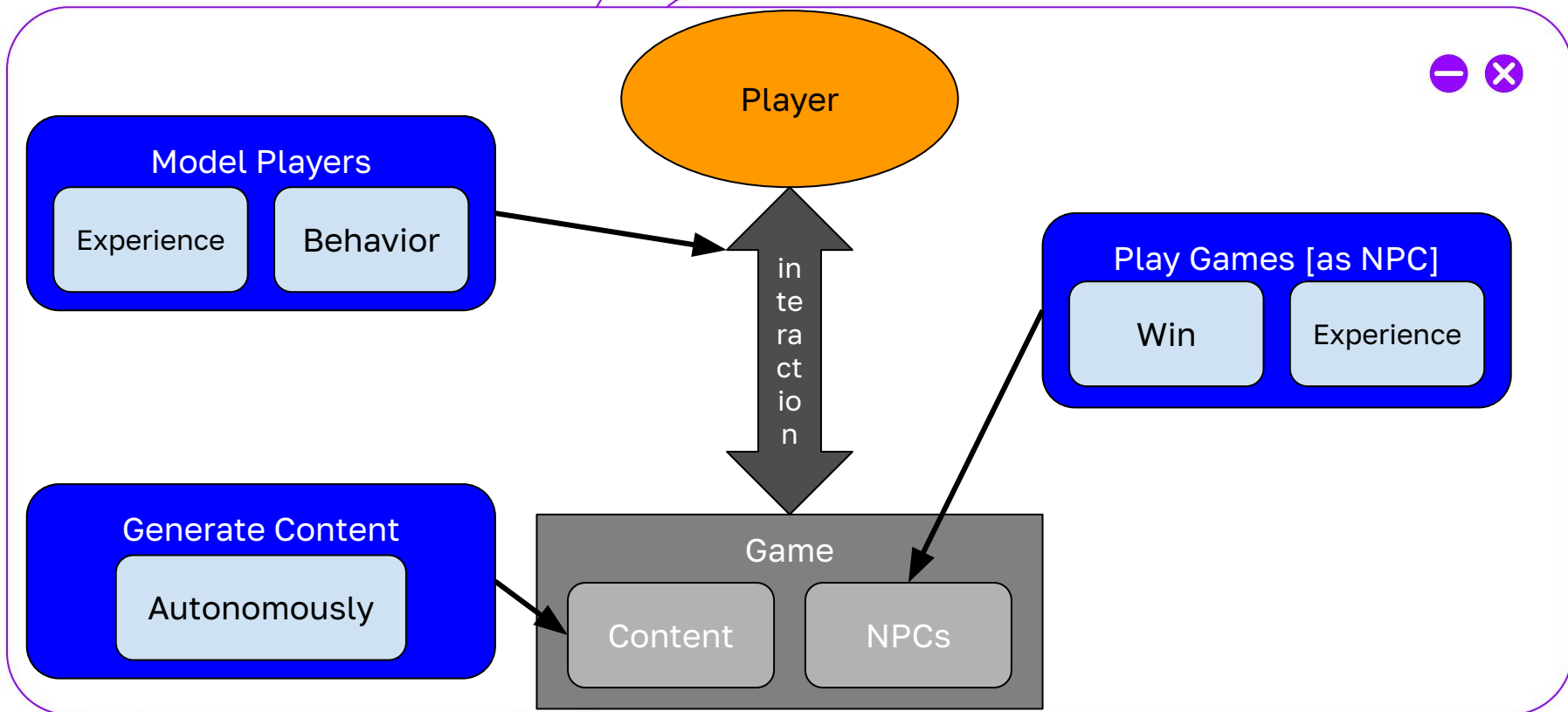


Стратегии и планирование

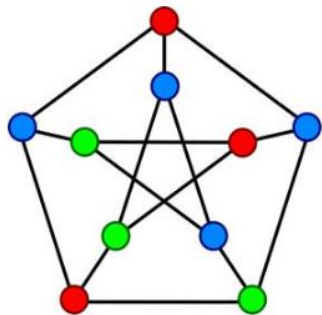


Генерация контента

ГЛАВА 1 _ Игровой ИИ: задачи



ГЛАВА 1 _ Игровой ИИ: подходы



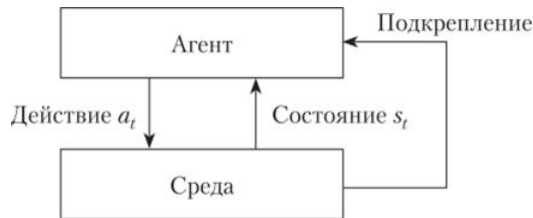
Алгоритмы, правила



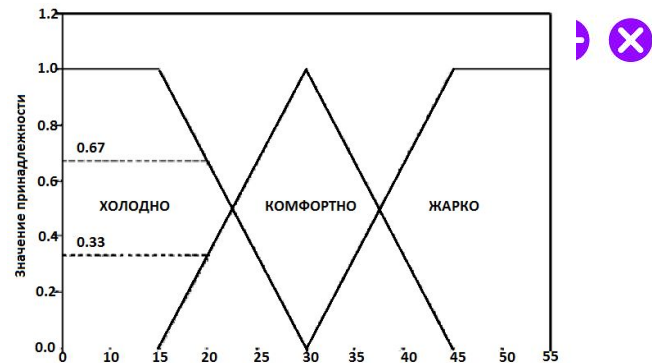
Эволюционные вычисления



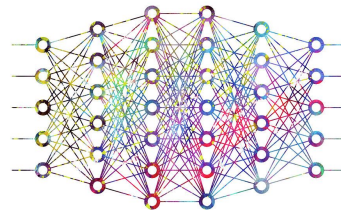
Прикладная Математика



Обучение с подкреплением



Нечеткость, логика и знания



Машинное обучение

ГЛАВА 1 _ Теория игр

«Камень - ножницы - бумага»

Два человека независимо друг от друга загадывают «камень», «ножницы», или «бумага». Камень тупит ножницы, ножницы режут бумагу, бумага побеждает камень. ($K > H > B > K$)

	камень	ножницы	бумага
камень	0	1	- 1
ножницы	- 1	0	1
бумага	1	- 1	0

«Семейный спор»

Семейная пара - муж и жена - решают вопрос, как проводить субботний вечер. У них существуют два вида развлечений: балет и футбол, жена предпочитает балет, муж - футбол.

Поскольку они привязаны друг к другу, максимальный выигрыш (4 единицы) каждый из них получает, если идет на любимое развлечение вместе с супругом.

Если они идут по одиночке, то получают по 2 единицы.

Если идут вместе на нелюбимое развлечение, то - по 1 единице.

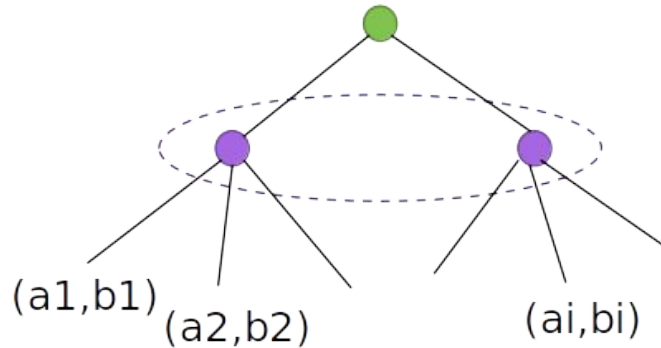
Если они попали на нелюбимое развлечение и по одному (жена на футболе, а муж - на балете), то выигрыша нет (0 единиц).

Муж Жена	Балет	Футбол
Балет	(4, 1)	(2, 2)
Футбол	(0, 0)	(1, 4)

**Игры в матричной
(нормальной) форме**

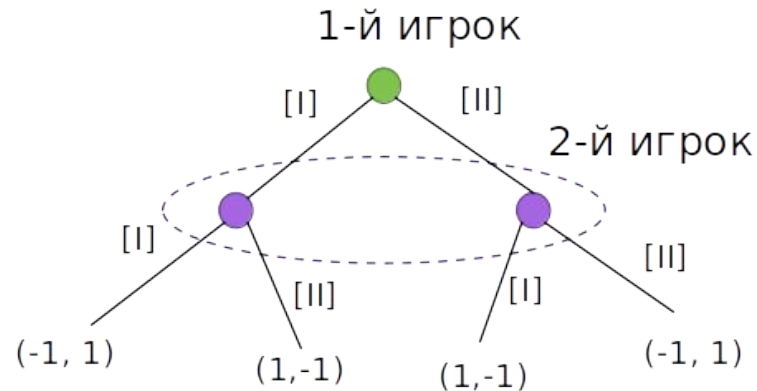
ГЛАВА 1 _ Теория игр

Игры в развернутой форме

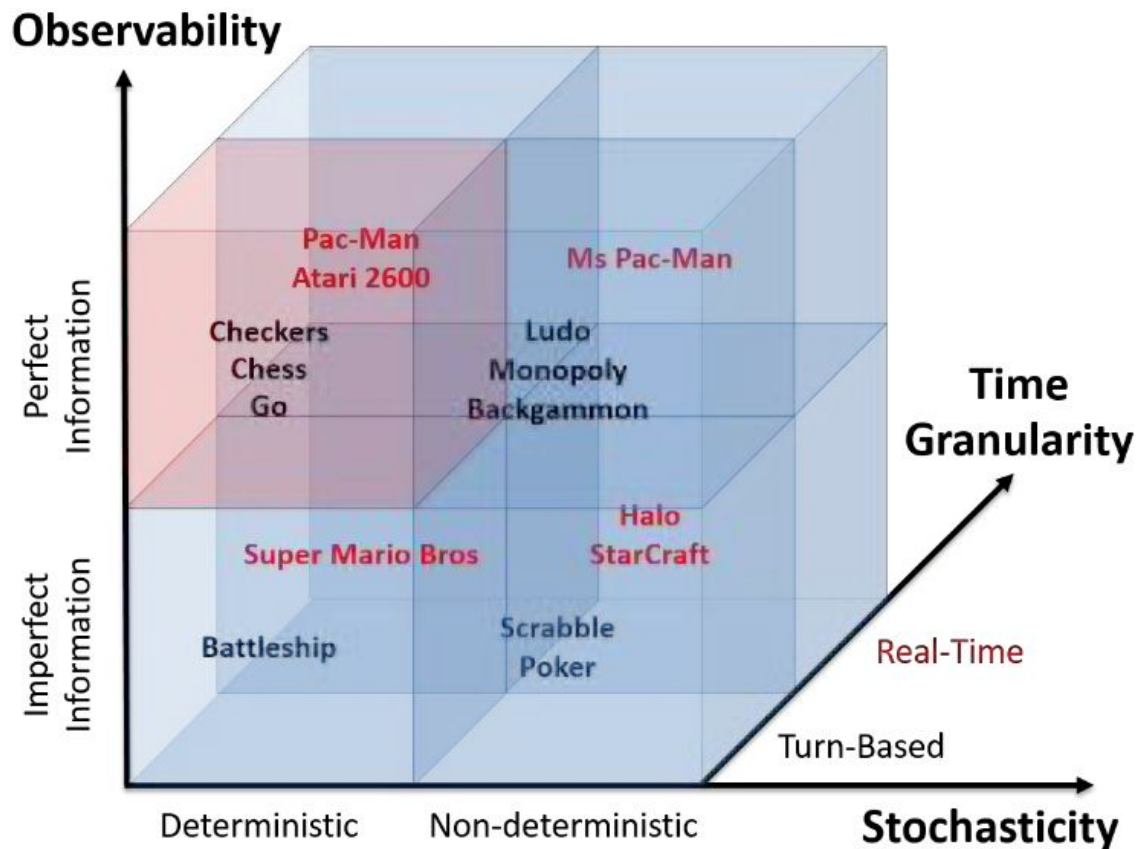


«Прятки»

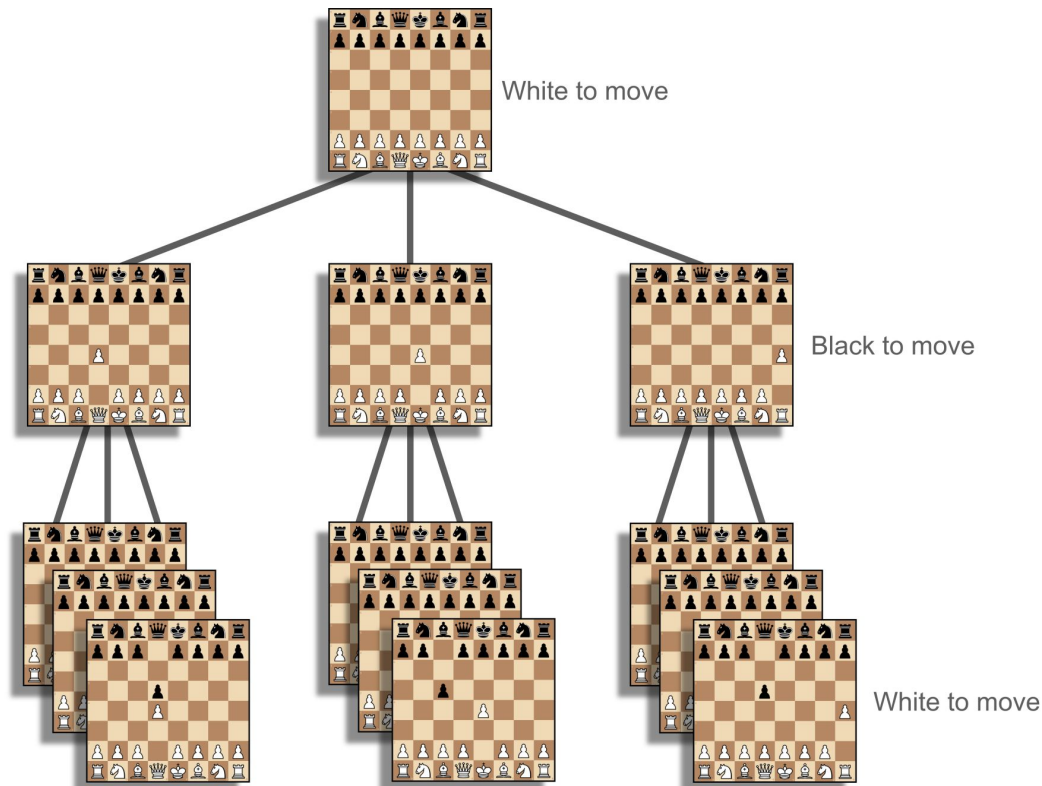
Первый может спрятаться в двух местах. Второй решает, где его искать (в первом или во втором месте). Если второй не находит, то выигрывает первый (1 у.е.), если находит – второй, а первый проигрывает (1 у.е.).



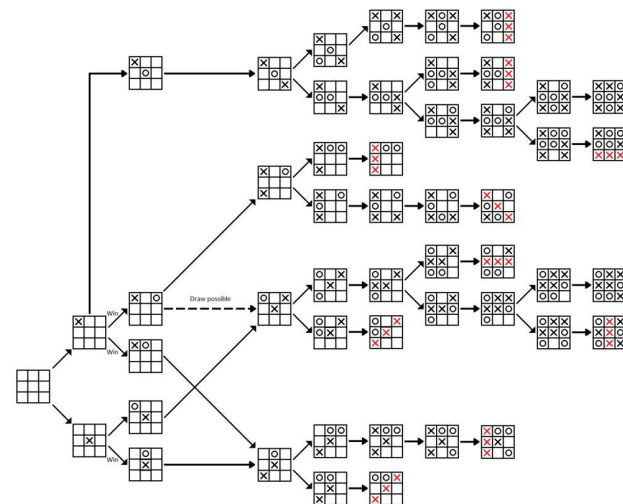
ГЛАВА 1 _ Теория игр



ГЛАВА 1 _ Теория игр



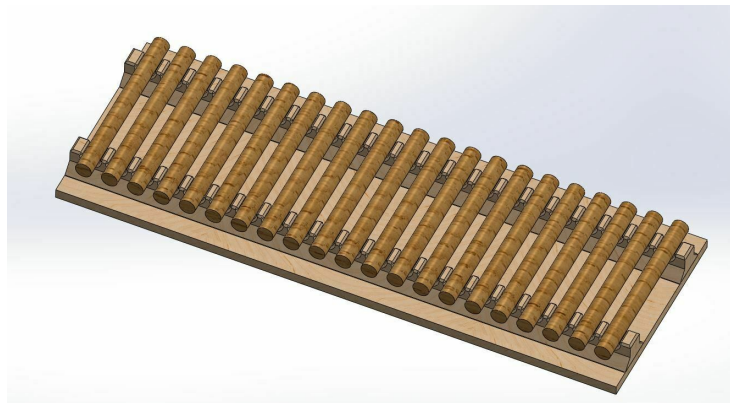
Игры в развернутой форме



ГЛАВА 1 _ Поиск по дереву

Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



Форт Боярд (телеигра)

ГЛАВА 1 _ Поиск по дереву

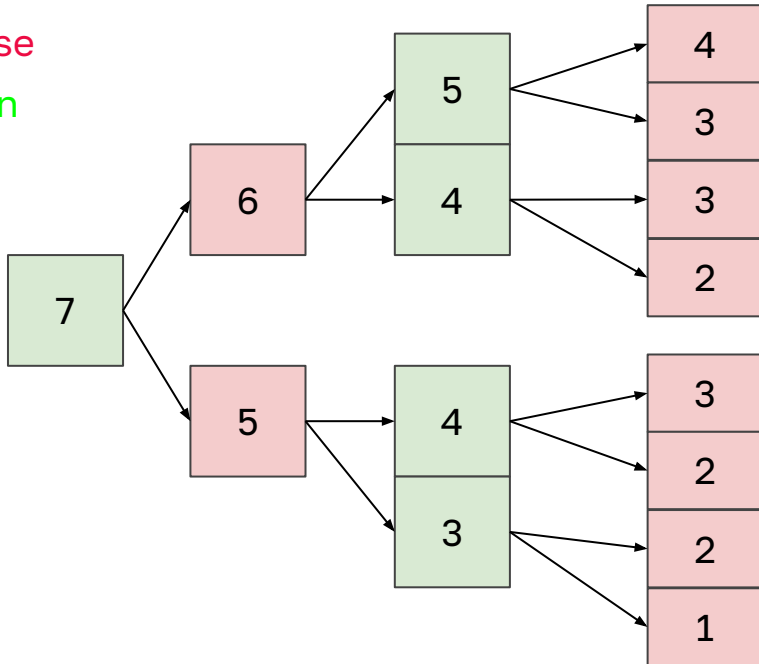
Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



Lose

Win



ГЛАВА 1 _ Поиск по дереву

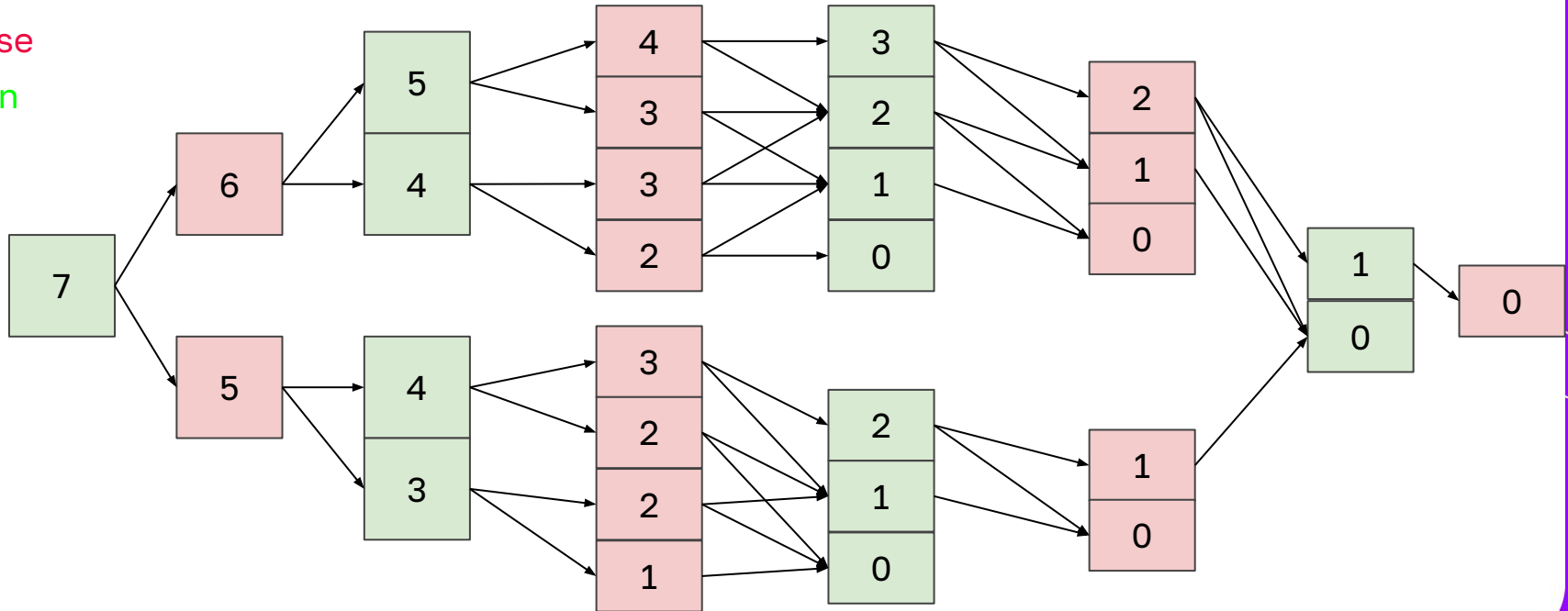
Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



Lose

Win



ГЛАВА 1 _ Поиск по дереву

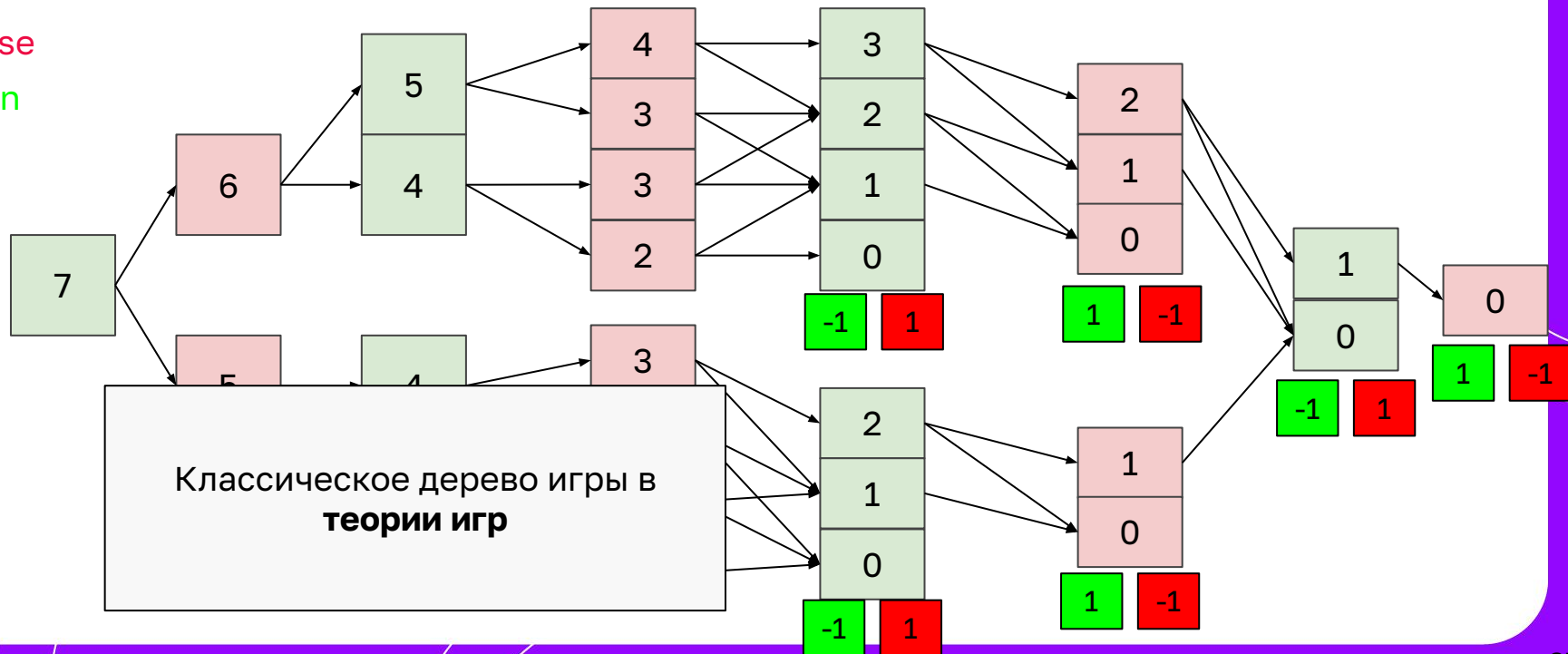
Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



Lose

Win



ГЛАВА 1 _ Поиск по дереву

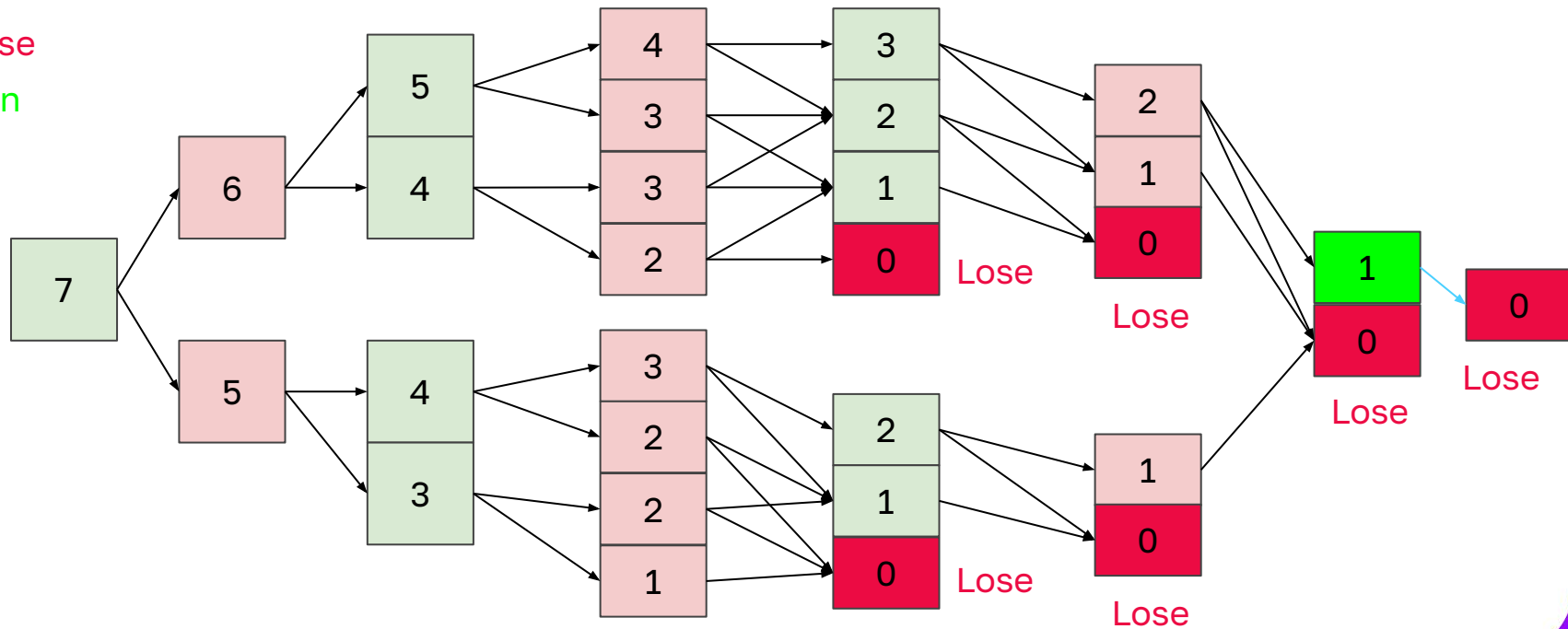
Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



Lose

Win



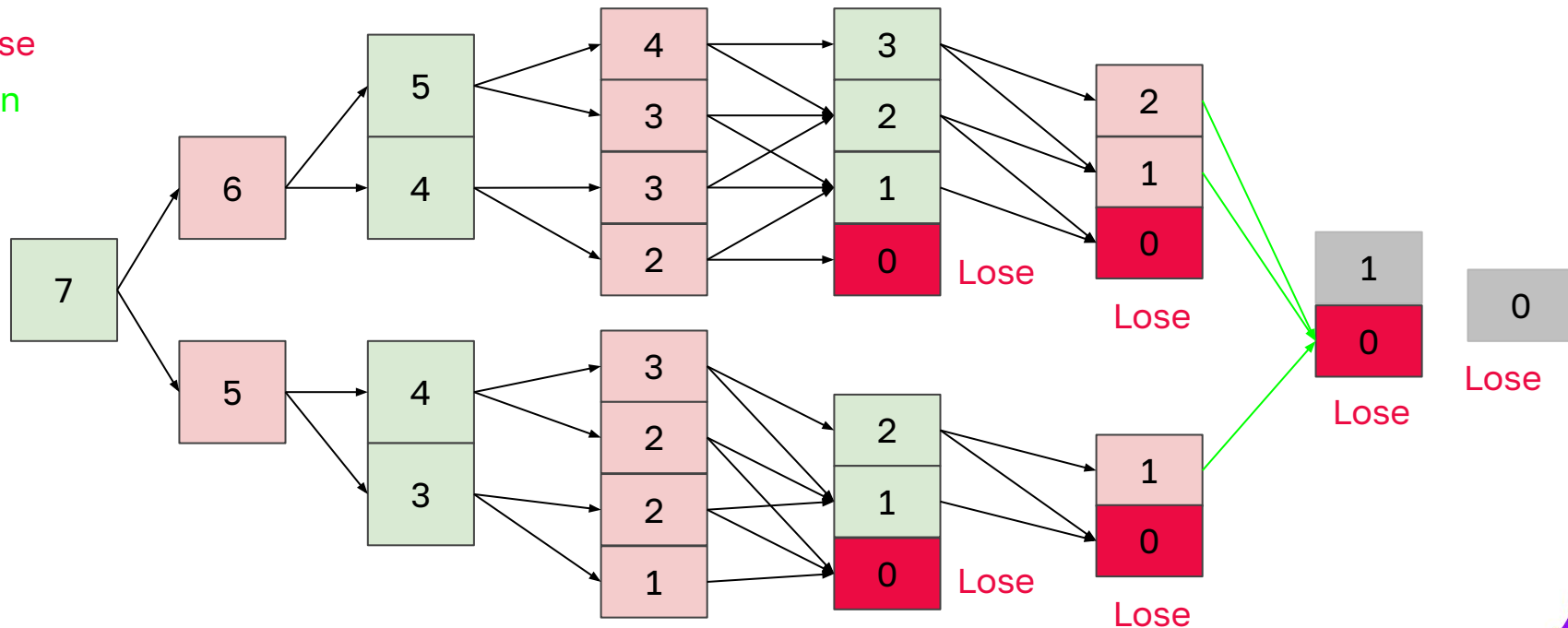
ГЛАВА 1 _ Поиск по дереву

Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



Lose
Win



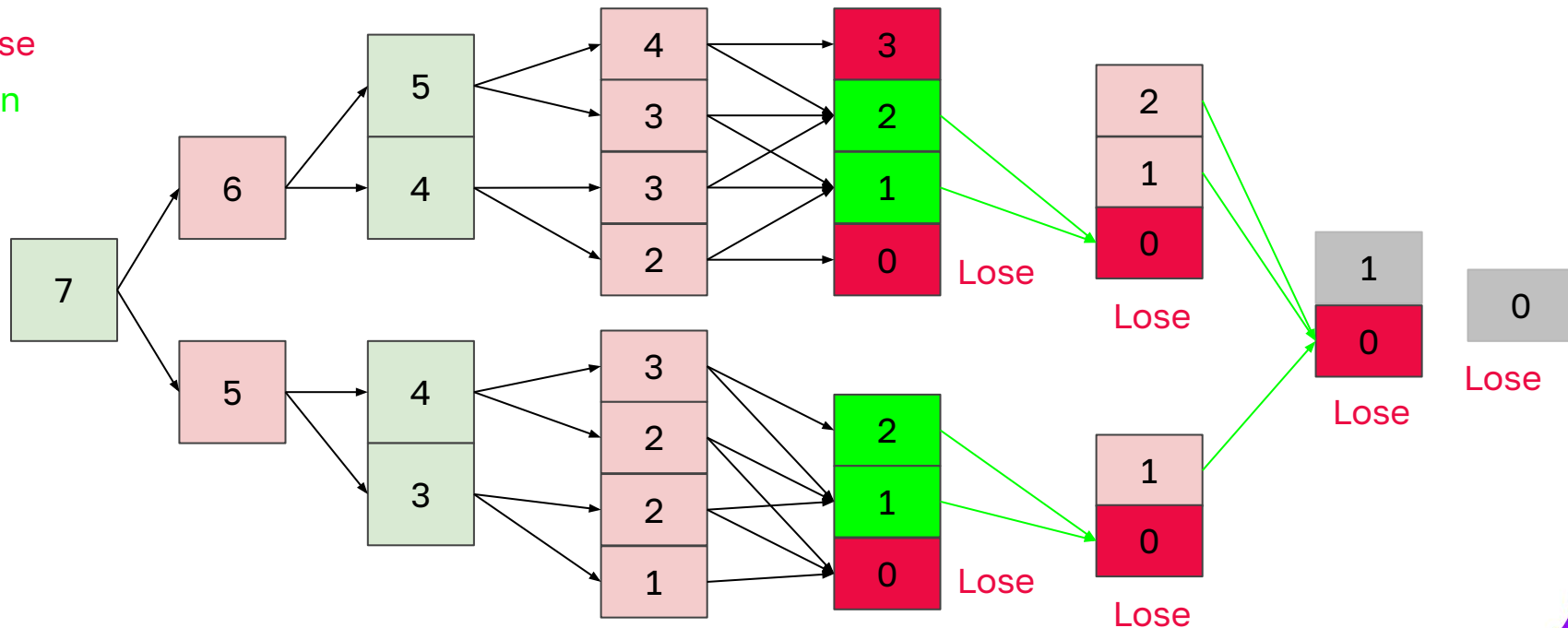
ГЛАВА 1 _ Поиск по дереву

Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



Lose
Win



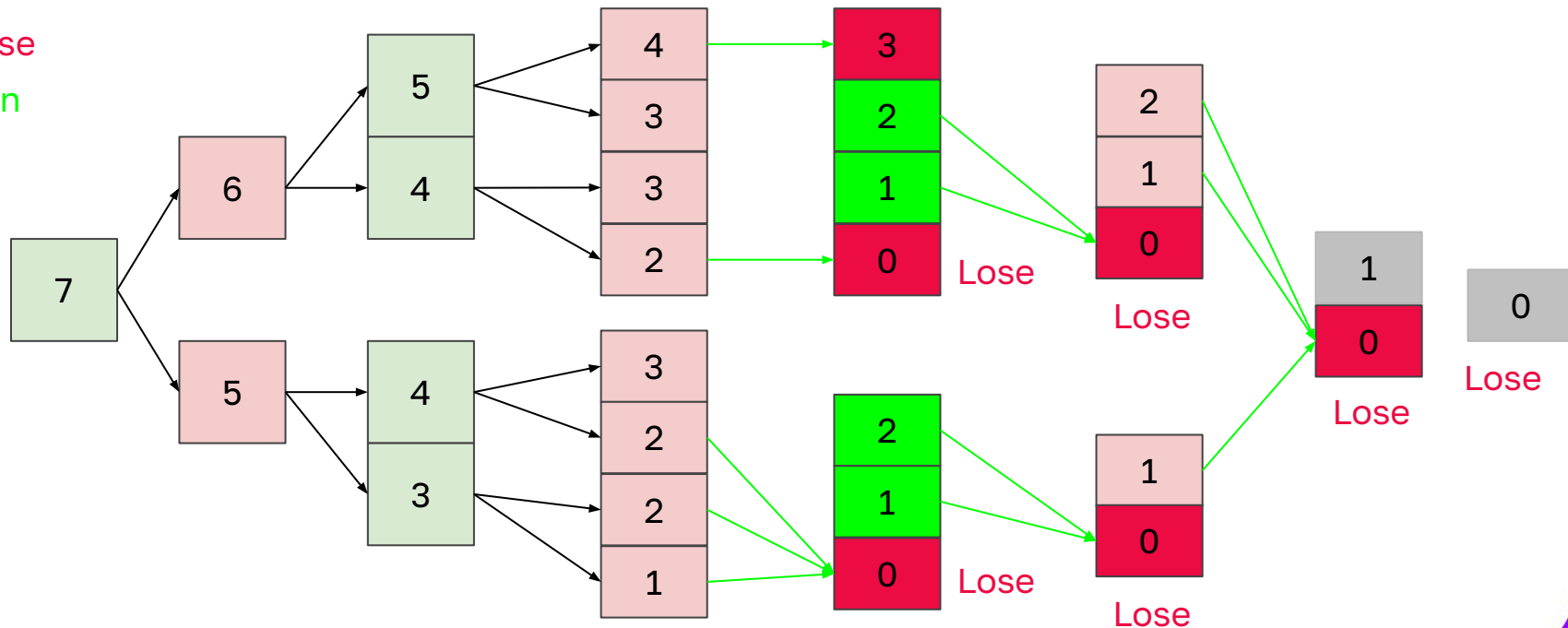
ГЛАВА 1 _ Поиск по дереву

Полный анализ дерева

Задача Баше-Менделеева (7 палочек), берем 1-2



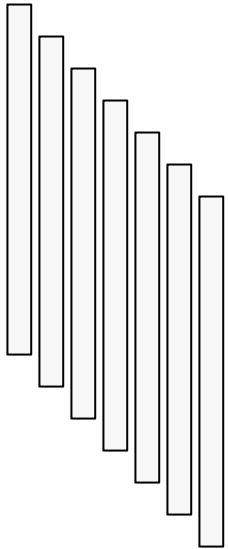
Lose
Win



ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

Задача Баше-Менделеева (7 палочек), берем 1-2

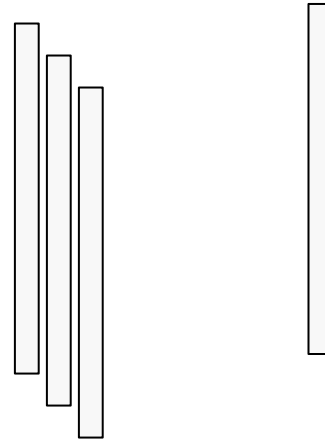
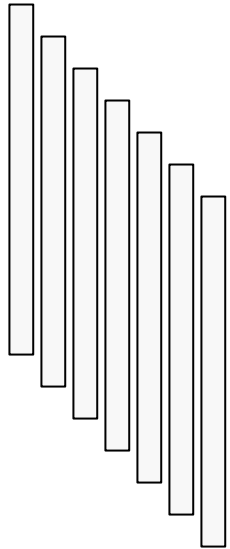


Победа: 1 или 2

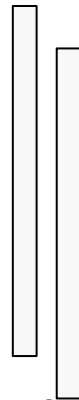
ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

Задача Баше-Менделеева (7 палочек), берем 1-2



Поражение

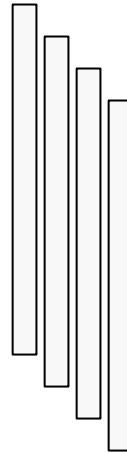
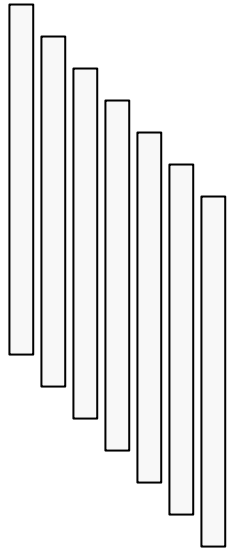


Победа: 1 или 2

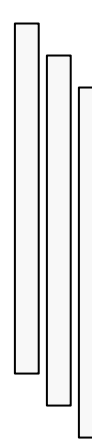
ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

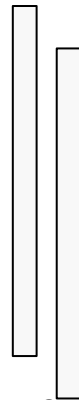
Задача Баше-Менделеева (7 палочек), берем 1-2



Победа



Поражение

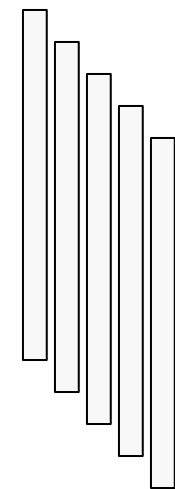
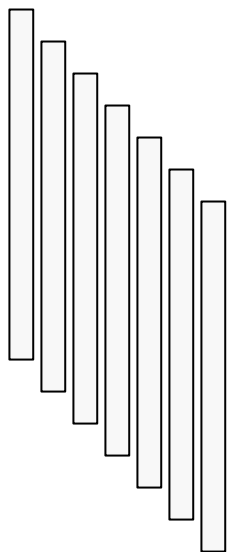


Победа: 1 или 2

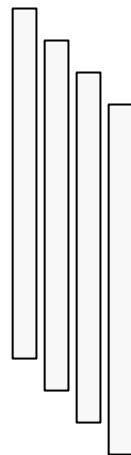
ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

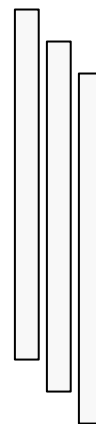
Задача Баше-Менделеева (7 палочек), берем 1-2



Победа



Победа



Поражение



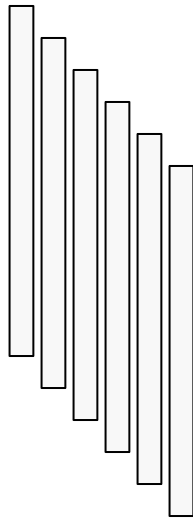
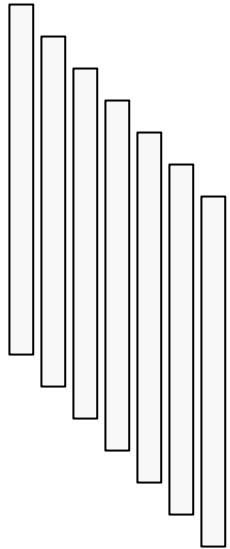
Победа: 1 или 2



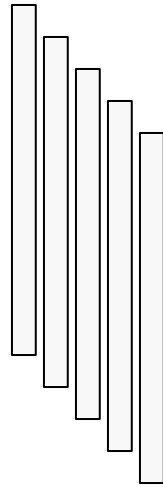
ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

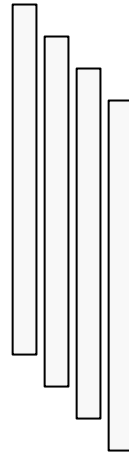
Задача Баше-Менделеева (7 палочек), берем 1-2



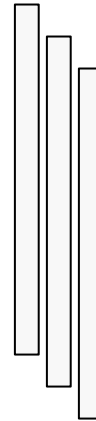
Поражение



Победа



Победа



Поражение

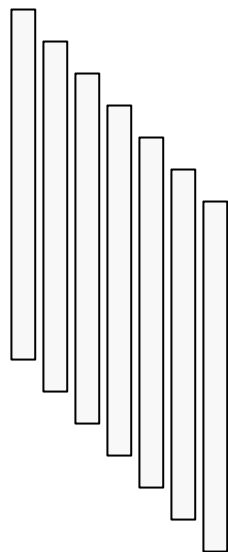


Победа: 1 или 2

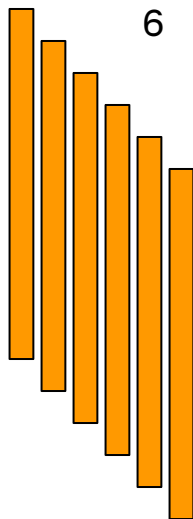
ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

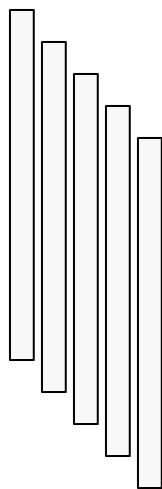
Задача Баше-Менделеева (7 палочек), берем 1-2



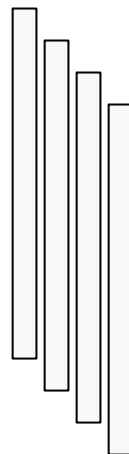
Победа



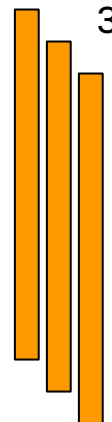
Поражение



Победа



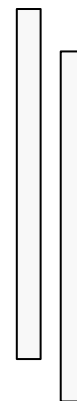
Победа



Поражение



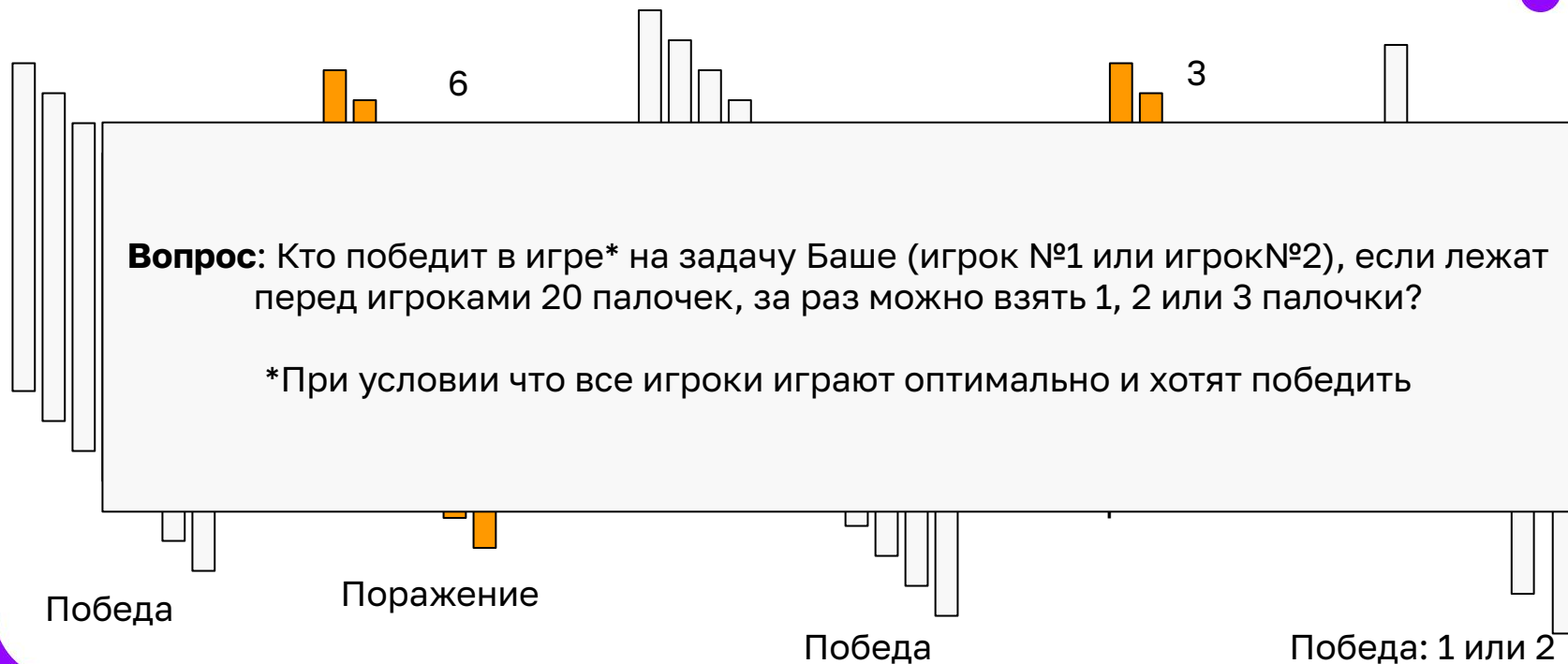
Победа: 1 или 2



ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

Задача Баше-Менделеева (7 палочек), берем 1-2



Анализ: Поиск оптимума

Задача Баше-Менделеева (7 палочек), берем 1-2

Вопрос: Кто победит в игре на задачу Баше (игрок №1 или игрок №2), если лежат перед игроками 20 палочек, за раз можно взять 1, 2 или 3 палочки?

Ответ:

1-2-3 – Победа

4 – Поражение

5-6-7 – Победа

8 – Поражение

9-10-11 – Победа

12 – Поражение

13-14-15 – Победа

16 – Поражение

17, 18, 19 – Победа

20 – Поражение

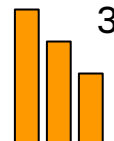
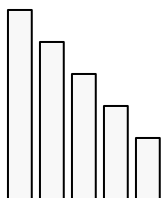
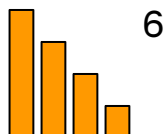
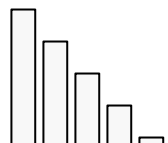
Победит Игрок №2.



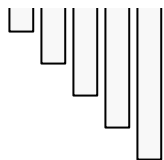
ГЛАВА 1 _ Поиск по дереву

Анализ: Поиск оптимума

Задача Баше-Менделеева (7 палочек), берем 1-2



Бери столько предметов, чтобы после твоего хода количество предметов было кратно $(M+1)$.

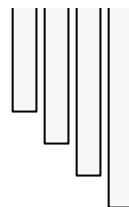


Победа



Поражение

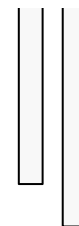
Победа



Победа



Поражение



Победа: 1 или 2

ГЛАВА 1 _ Поиск по дереву

Что же делать с такими играми, как шахматы? Или шашки?



Алгоритм МинМакс (МинМакс процедура)



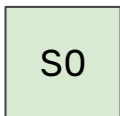
1. Просматриваем (строим) часть дерева игры от нашей ситуации.

Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

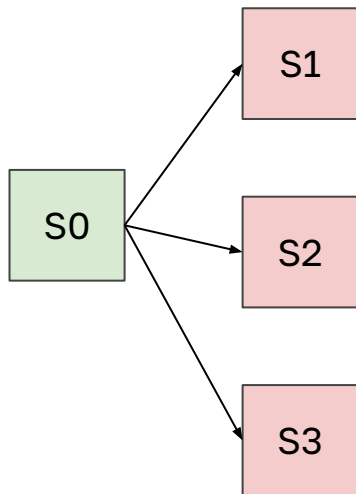


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

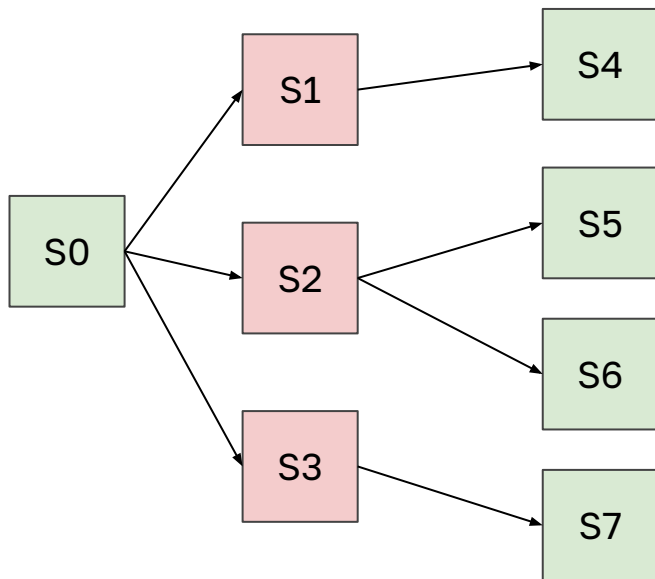


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

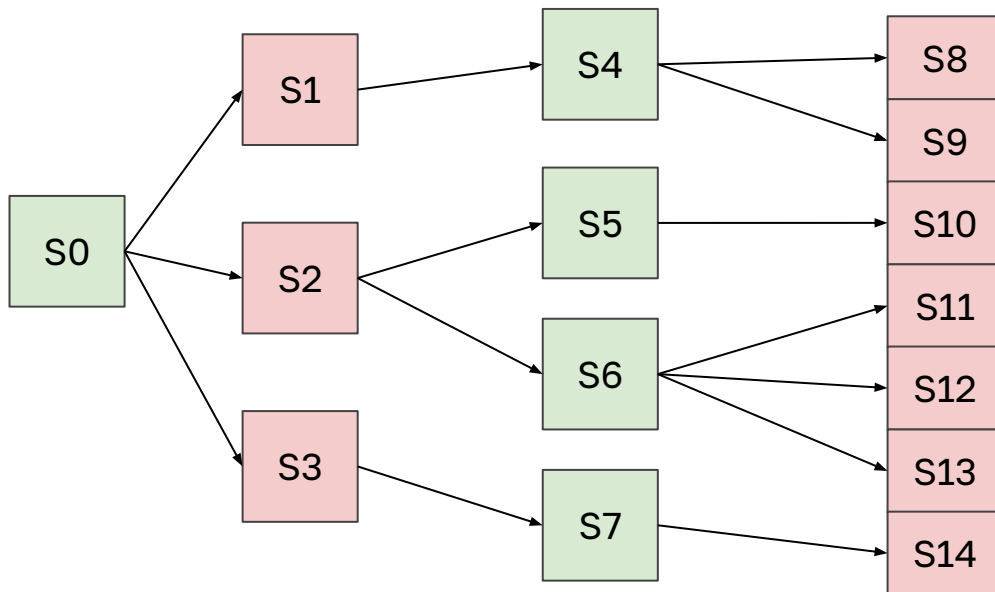


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

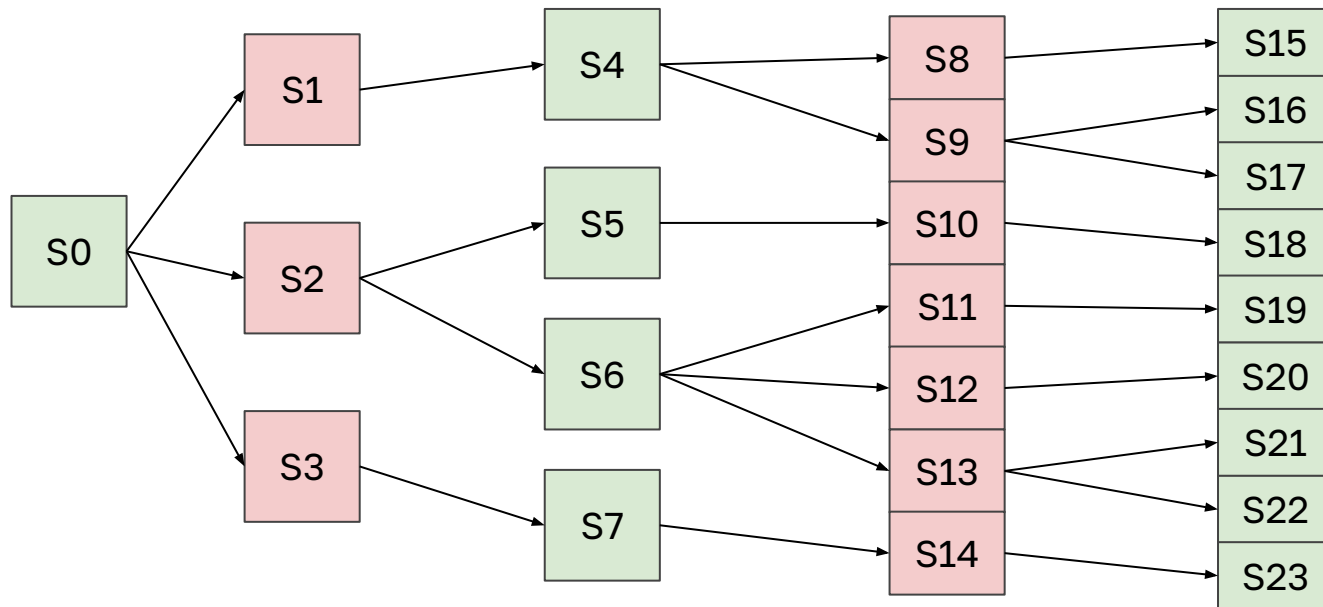


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

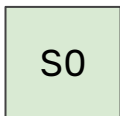


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

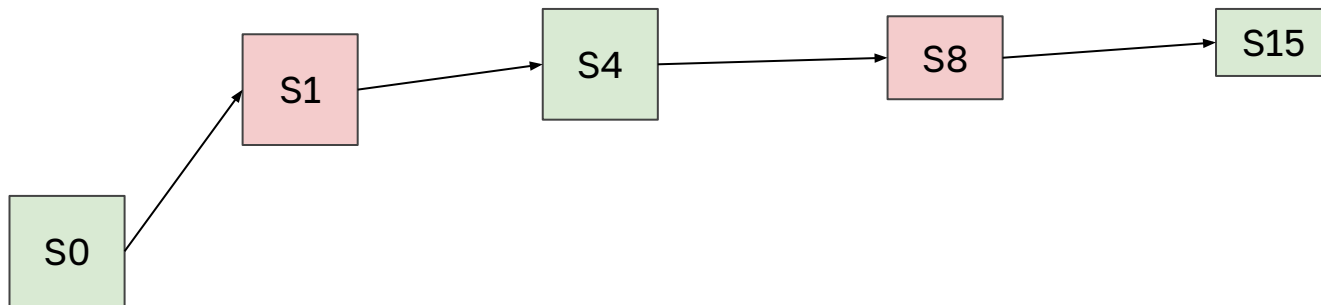


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

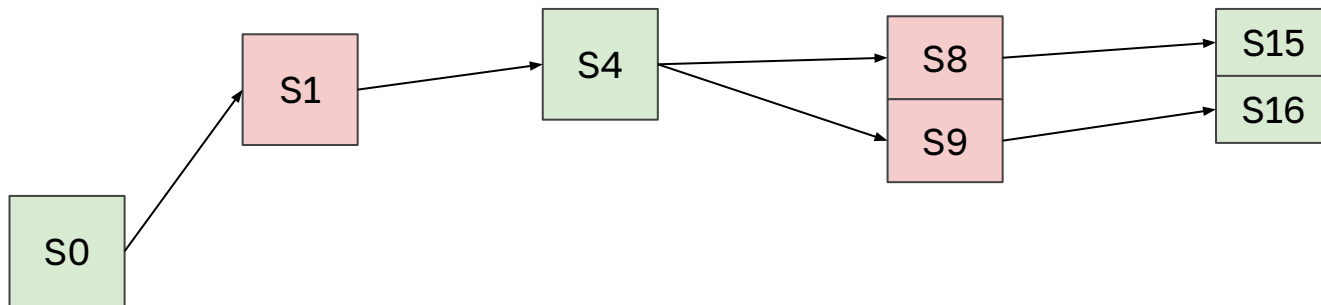


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

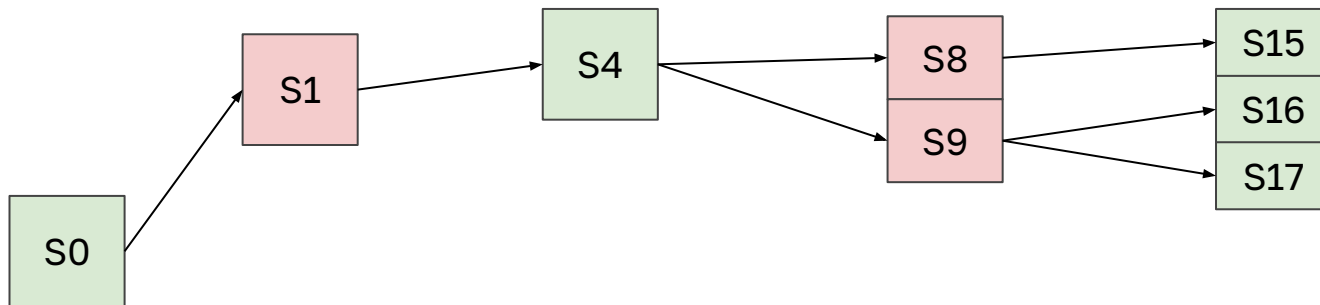


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

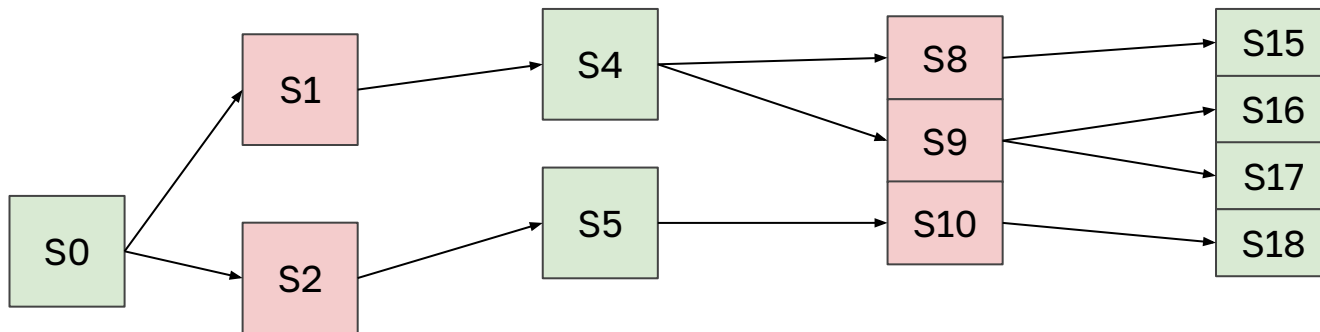


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

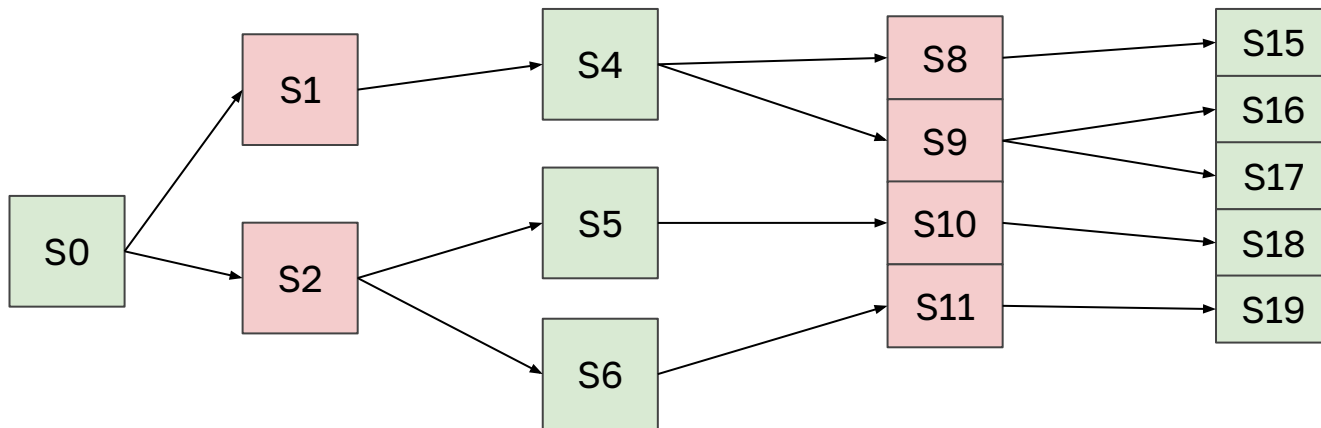


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

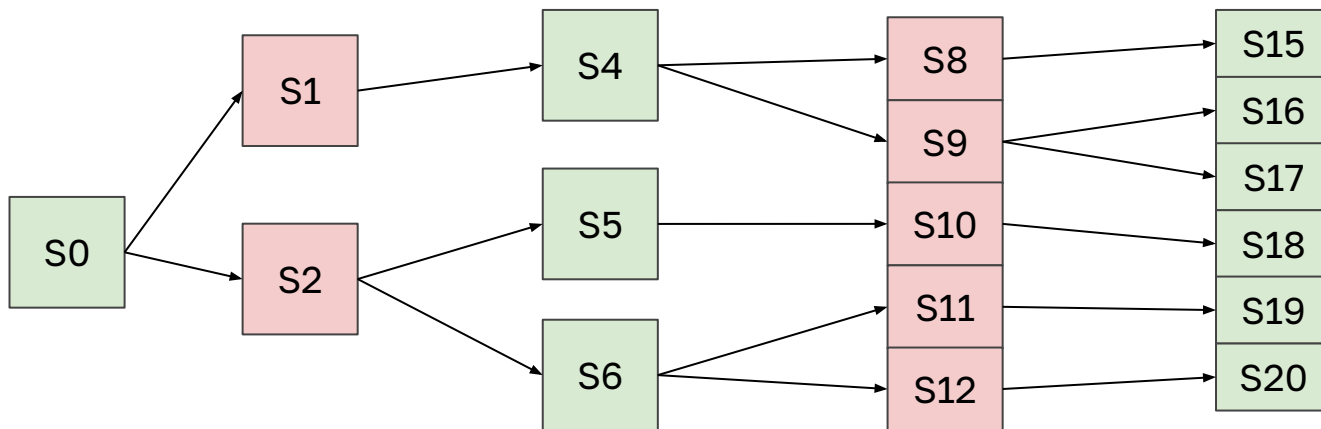


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

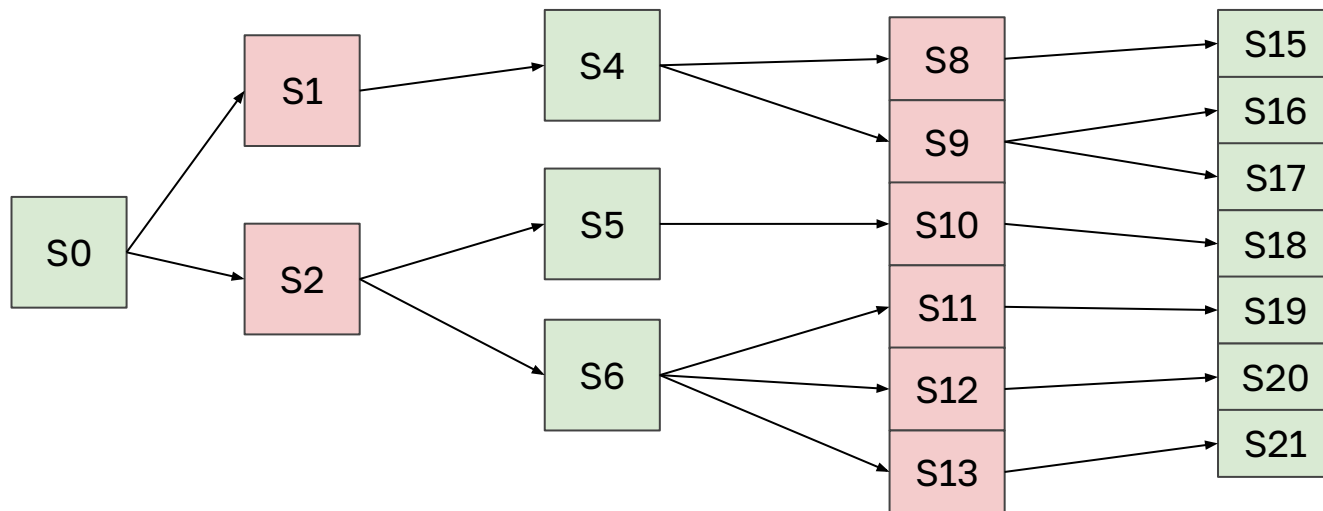


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

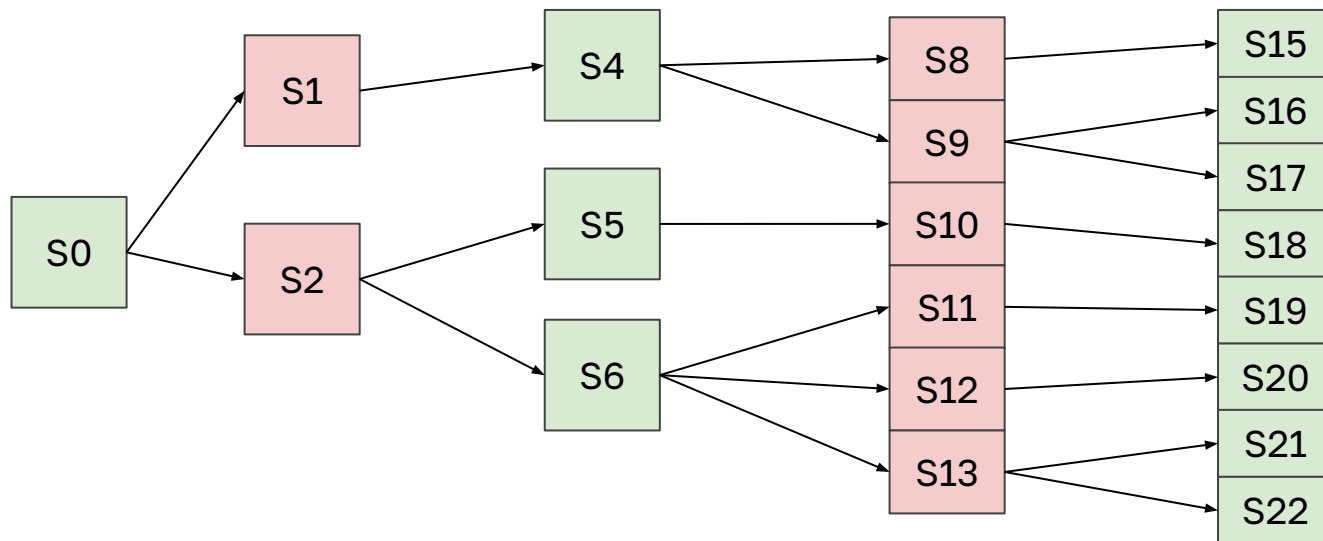


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

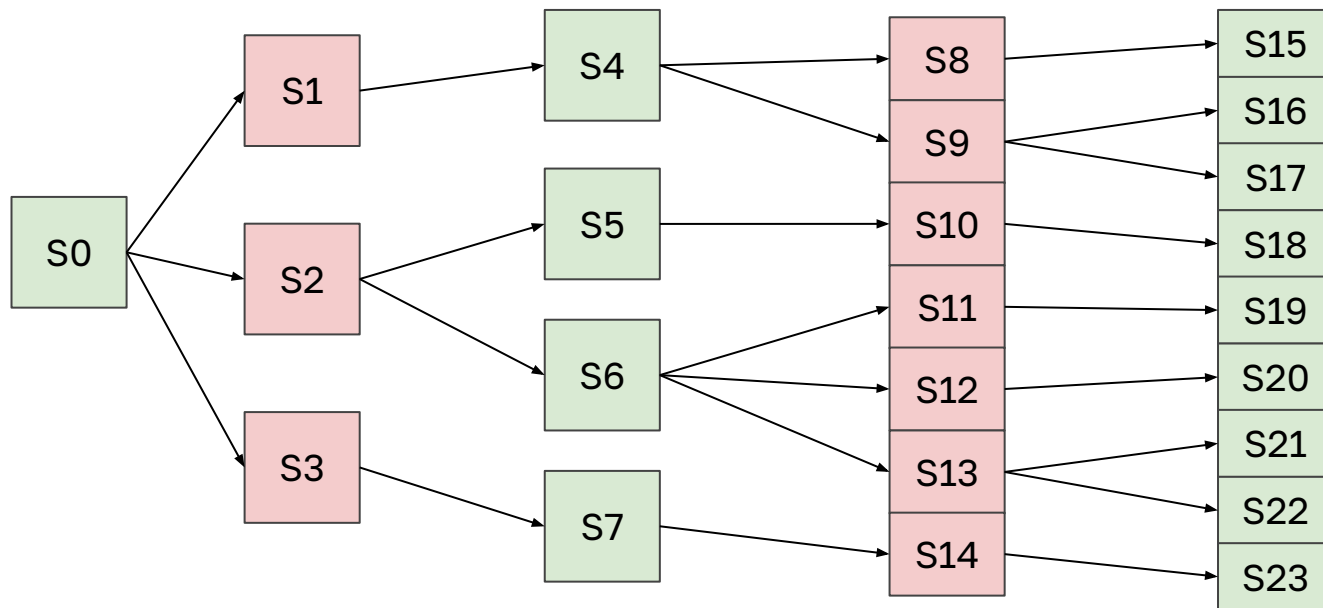


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



1. Просматриваем (строим) часть дерева игры от нашей ситуации.

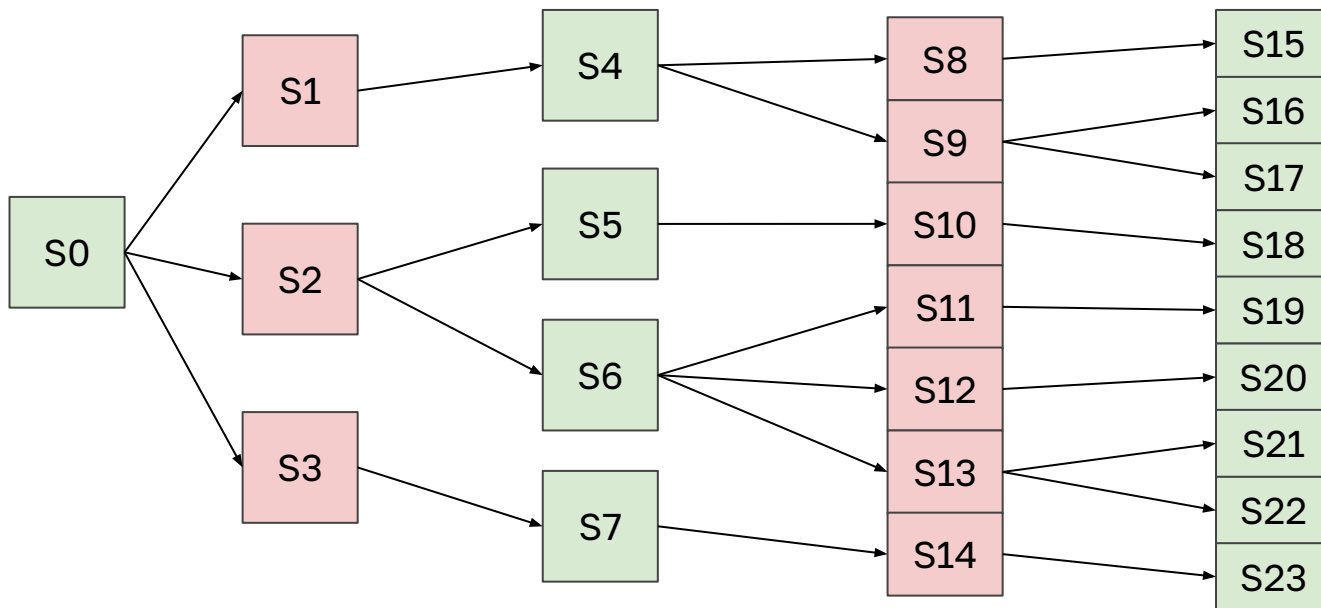


Искусственное
окончание
перебора
вершин в
игровом
дереве:
например,
время
перебора или
же глубина
поиска.

Алгоритм МинМакс (МинМакс процедура)



2. Применяем статическую оценочную функцию (СОФ)

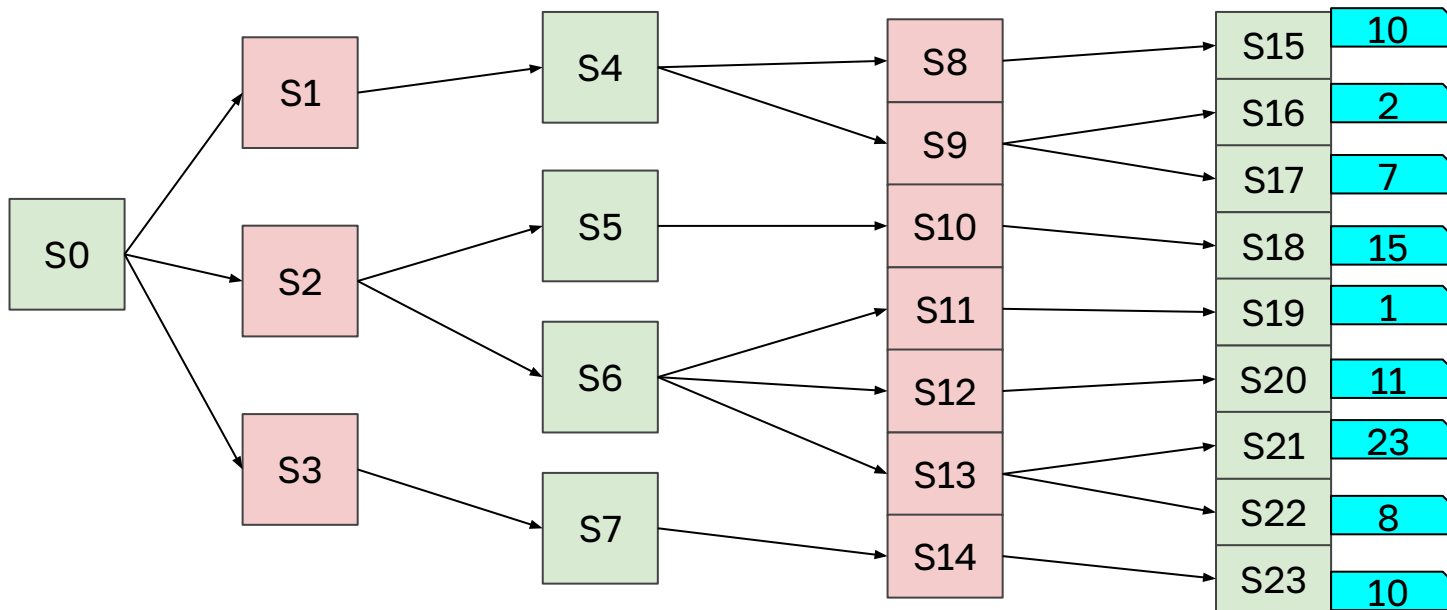


СОФ -
эвристика,
оценивающая
состояние
игры.
Например,
дамки, удар
или не удар
короля или
ладьи,
расположение
ноликов

Алгоритм МинМакс (МинМакс процедура)



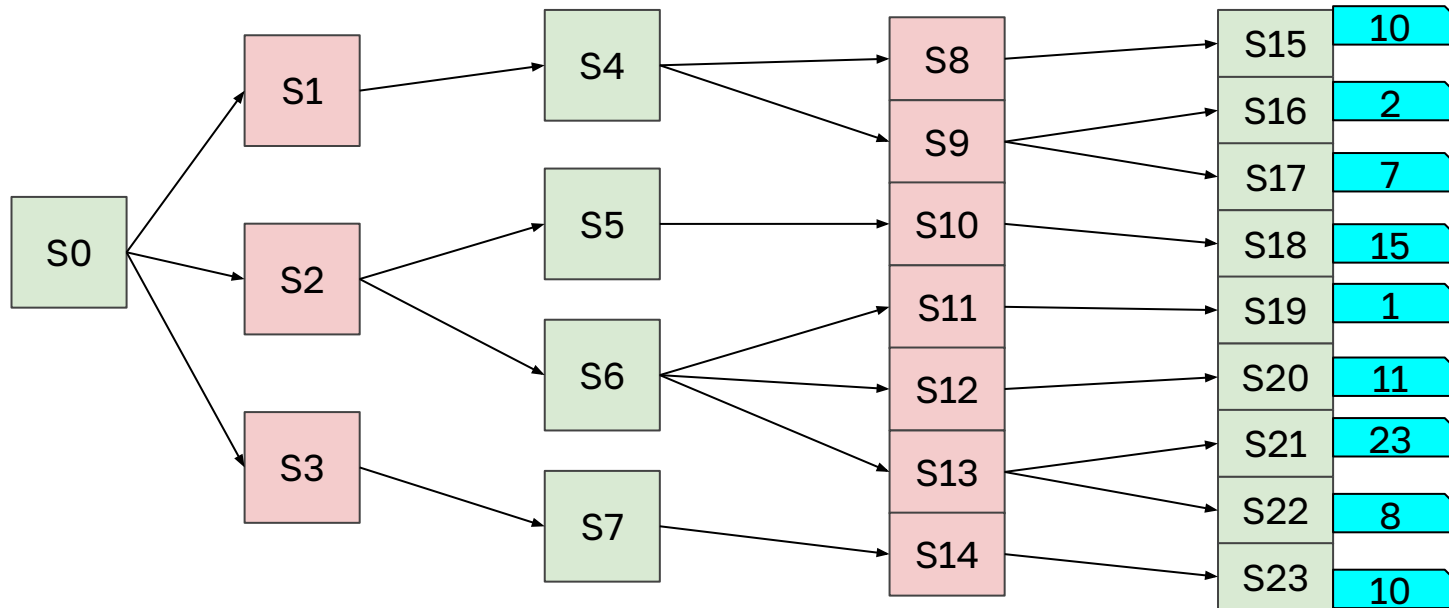
2. Применяем статическую оценочную функцию (СОФ)



Алгоритм МинМакс (МинМакс процедура)



3. Применяем минимаксную процедуру



МАКС

МИН

МАКС

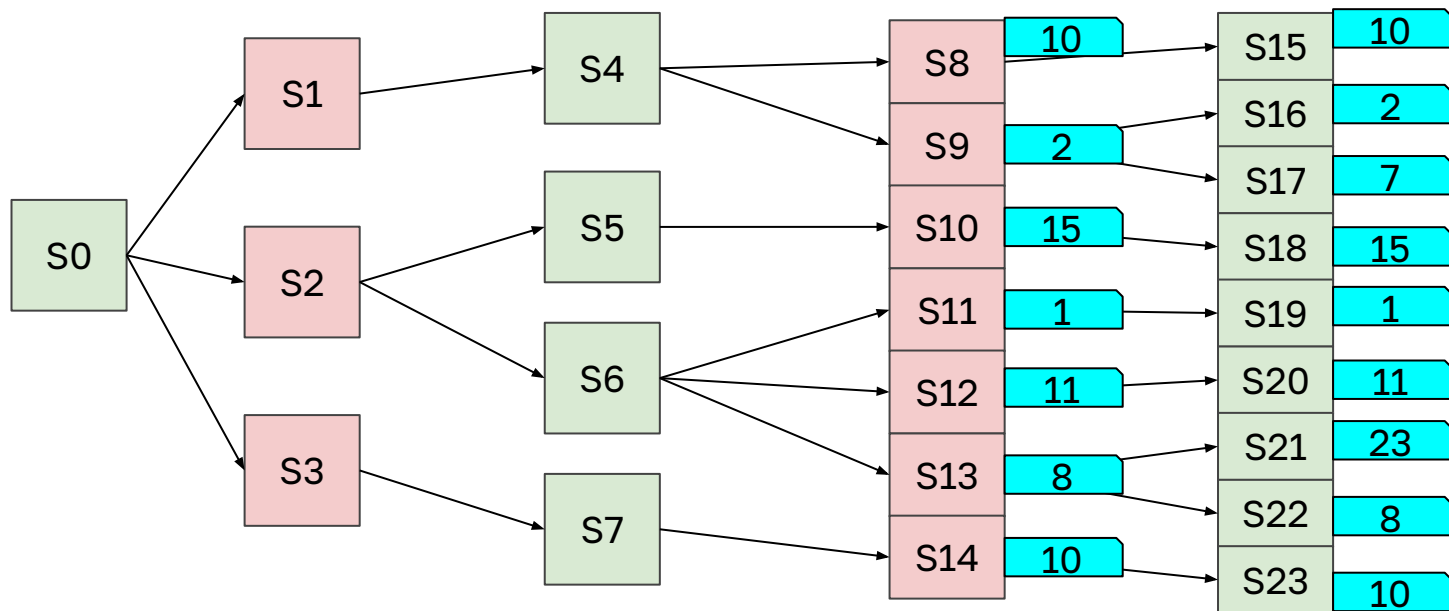
МИН

МАКС

Алгоритм МинМакс (МинМакс процедура)



3. Применяем минимаксную процедуру



МАКС

МИН

МАКС

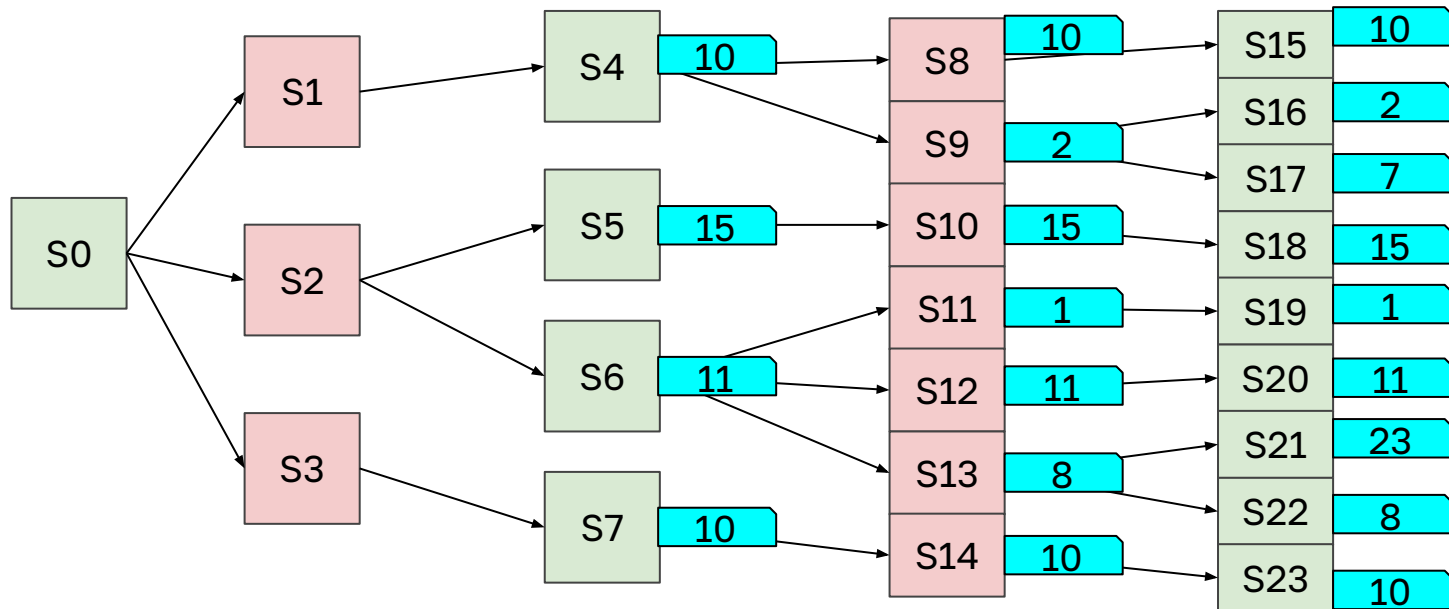
МИН

МАКС

Алгоритм МинМакс (МинМакс процедура)



3. Применяем минимаксную процедуру



МАКС

МИН

МАКС

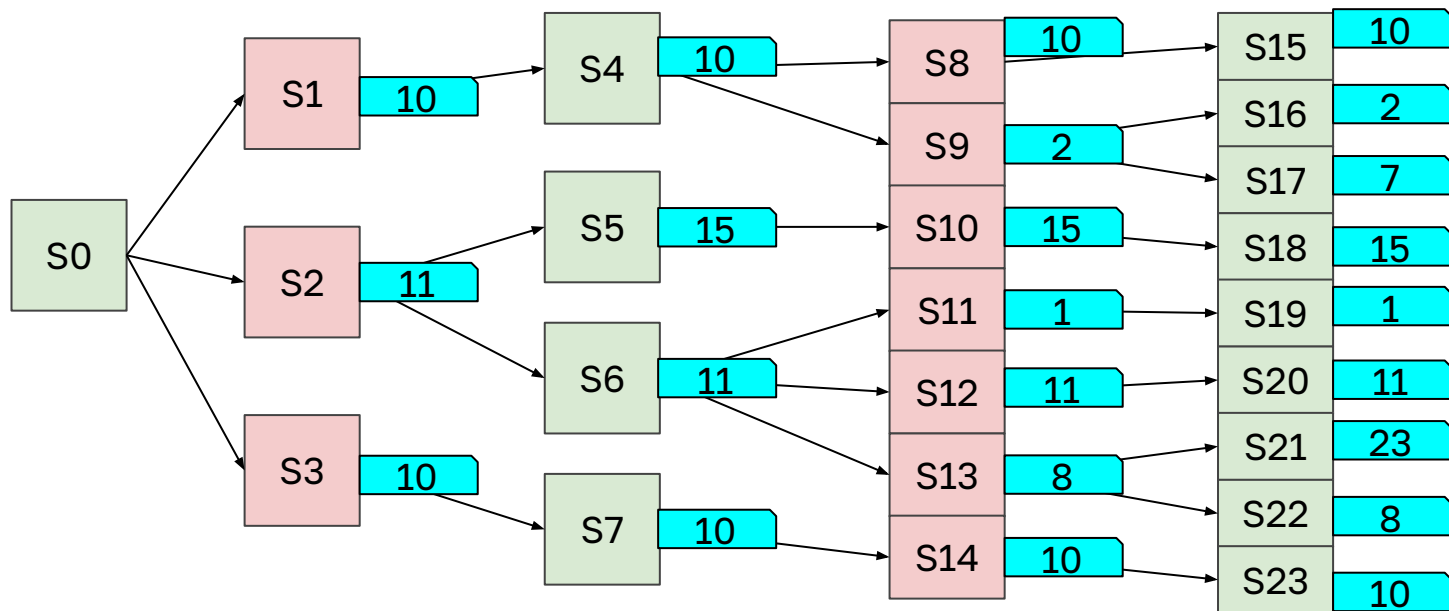
МИН

МАКС

Алгоритм МинМакс (МинМакс процедура)



3. Применяем минимаксную процедуру



МАКС

МИН

МАКС

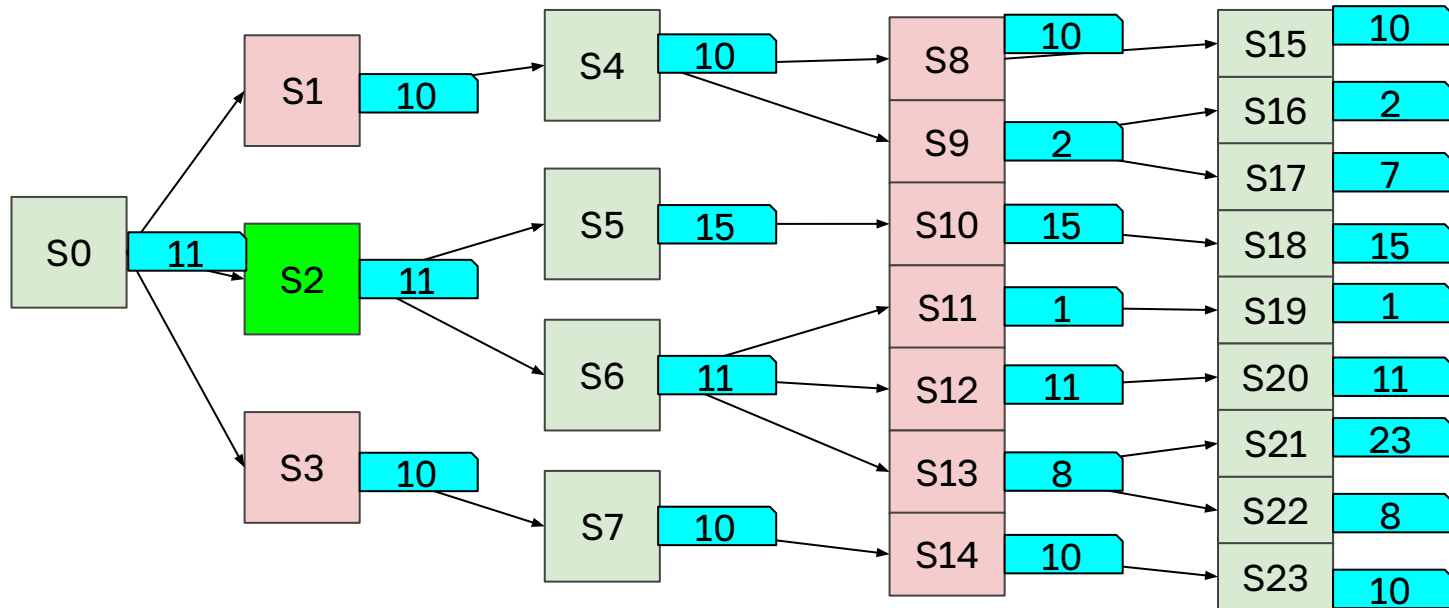
МИН

МАКС

Алгоритм МинМакс (МинМакс процедура)



3. Применяем минимаксную процедуру



МАКС

МИН

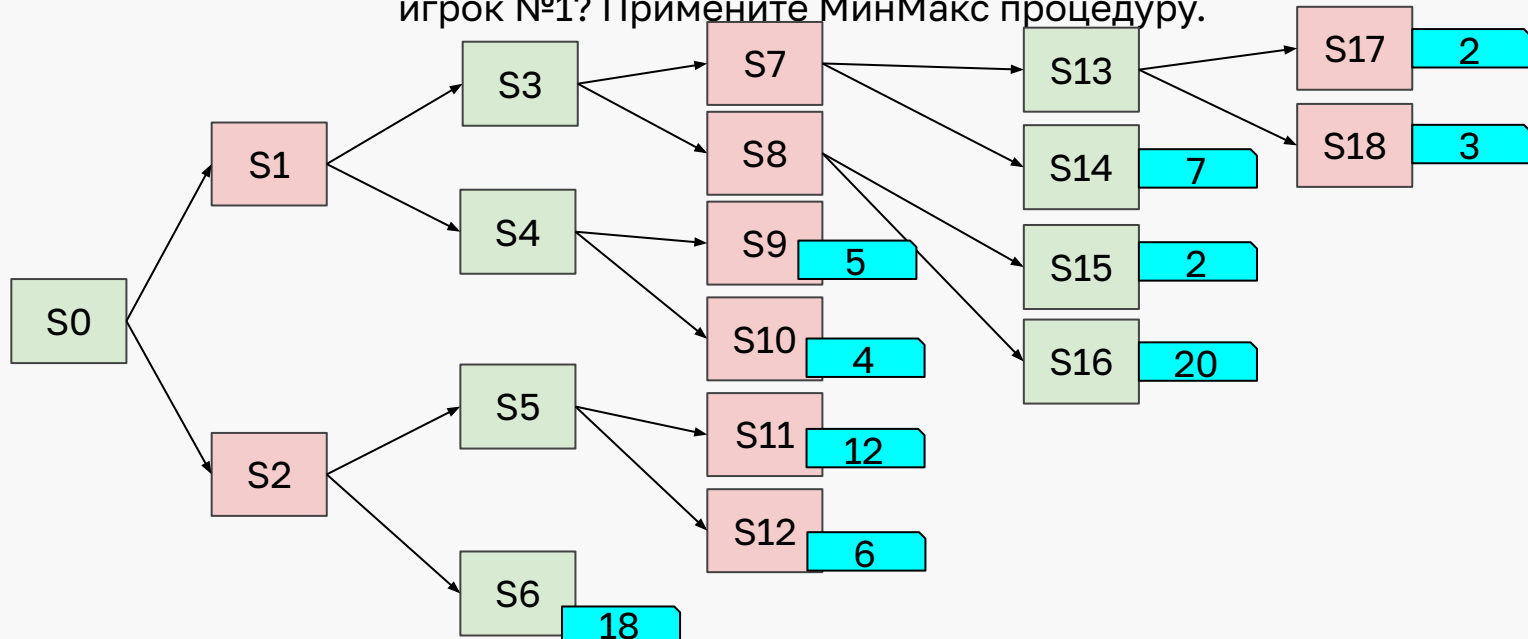
МАКС

МИН

МАКС

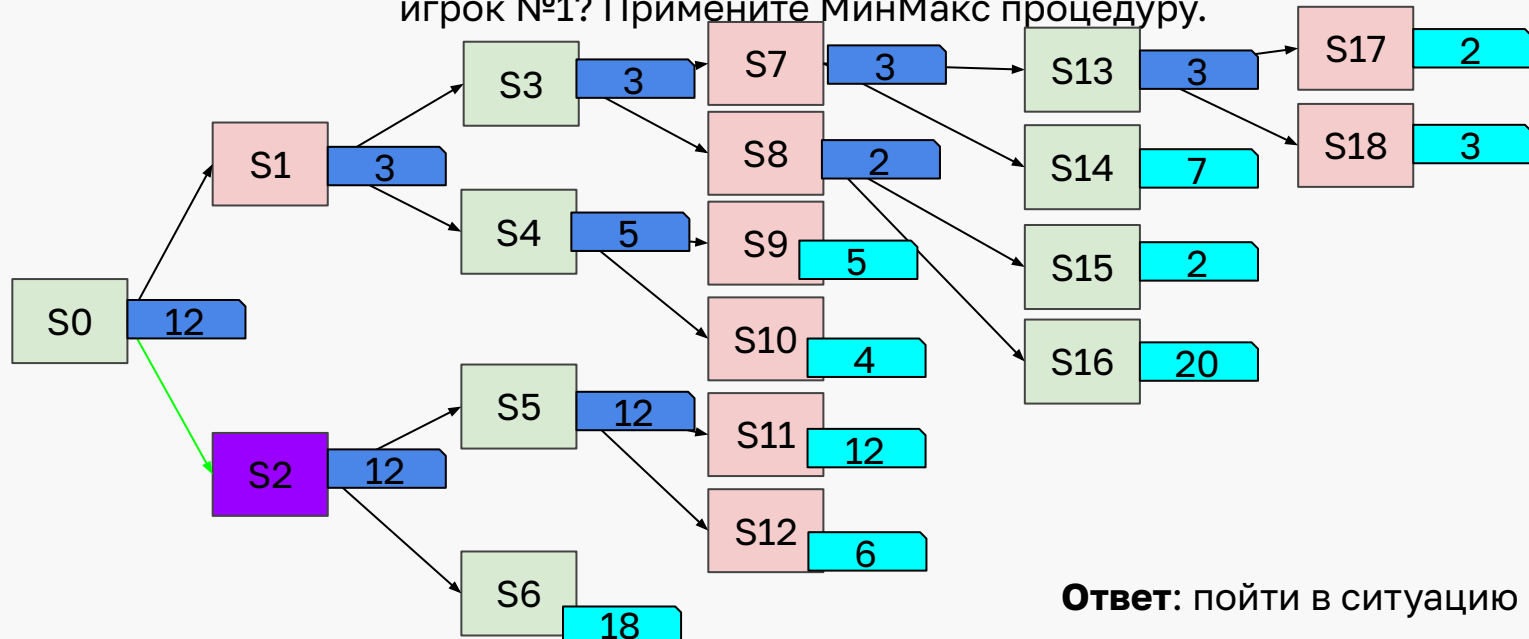
Алгоритм МинМакс (МинМакс процедура)

Задача. Перед Вами задана часть дерева игры и подсчитанные СОФ. Как должен сходить игрок №1? Примените МинМакс процедуру.



Алгоритм МинМакс (МинМакс процедура)

Задача. Перед Вами задана часть дерева игры и подсчитанные СОФ. Как должен сходить игрок №1? Примените МинМакс процедуру.



Алгоритм МинМакс (МинМакс процедура)



3. Применяем минимаксную процедуру

Считается, что оценки, полученные с помощью **минимаксной процедуры**, есть более **надежные** меры относительного достоинства промежуточных вершин, чем оценки, полученные прямым применением статической оценочной функции.



МАКС

МИН

МАКС

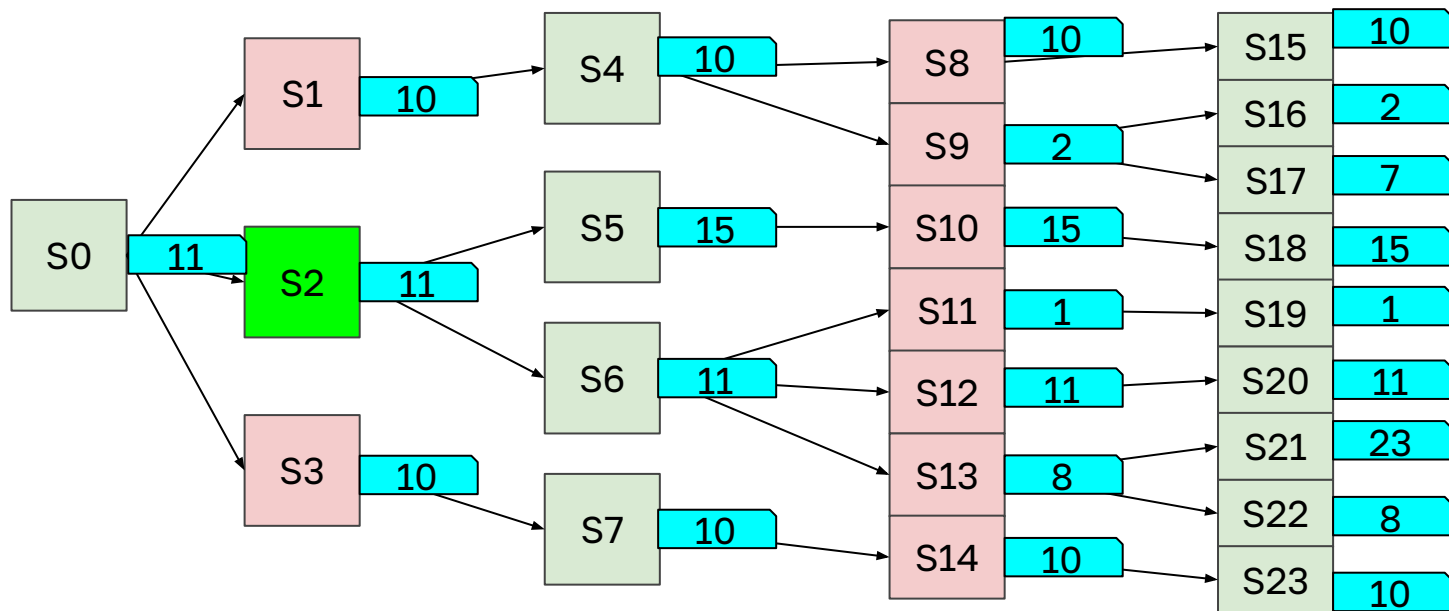
МИН

МАКС

Алгоритм МинМакс (МинМакс процедура)



3. Применяем минимаксную процедуру



МАКС

МИН

МАКС

МИН

МАКС

Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



Минимаксная процедура **неэффективная** стратегия поиска хорошего хода. Чтобы сделать процедуру более экономной, необходимо вычислять статические оценки концевых вершин и минимаксные оценки промежуточных вершин одновременно с построением игрового дерева

Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



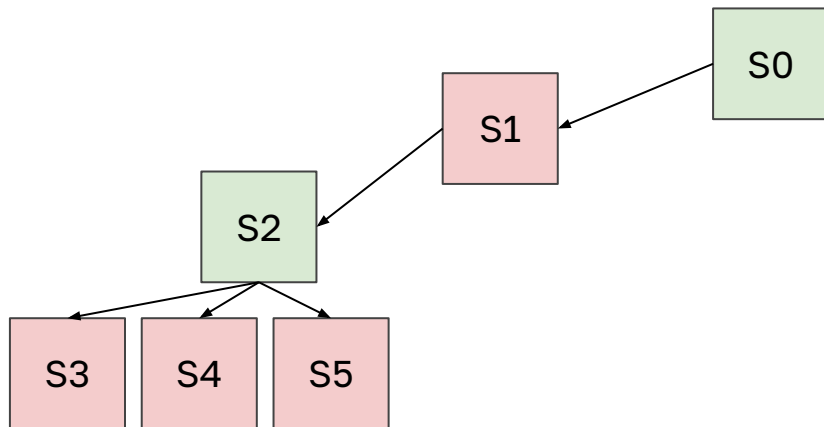
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.

S0

Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



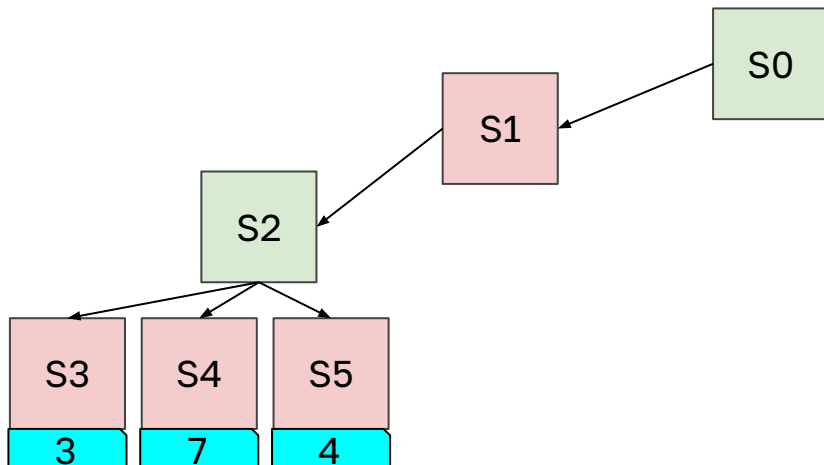
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



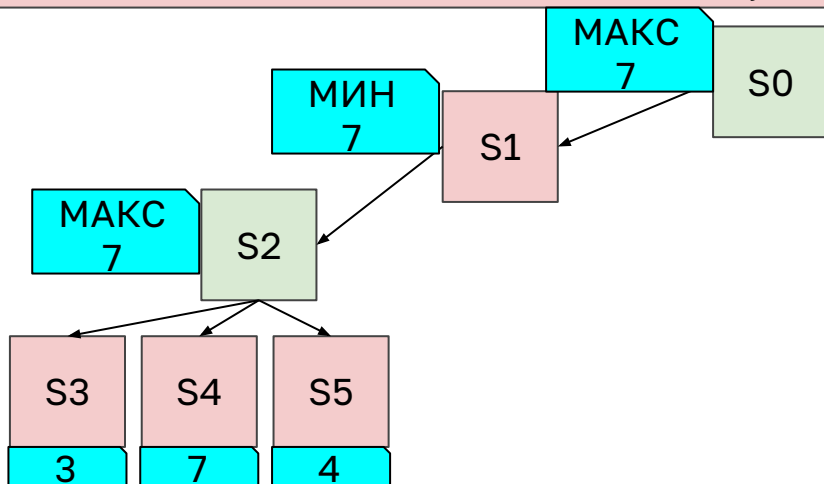
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



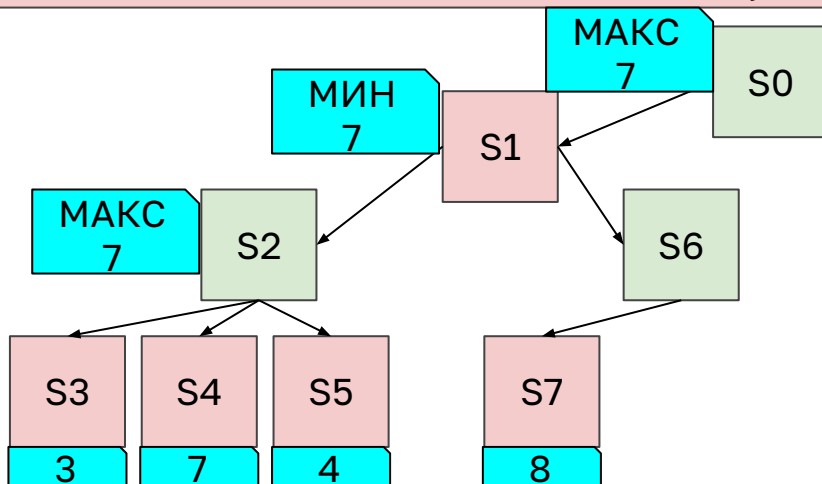
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



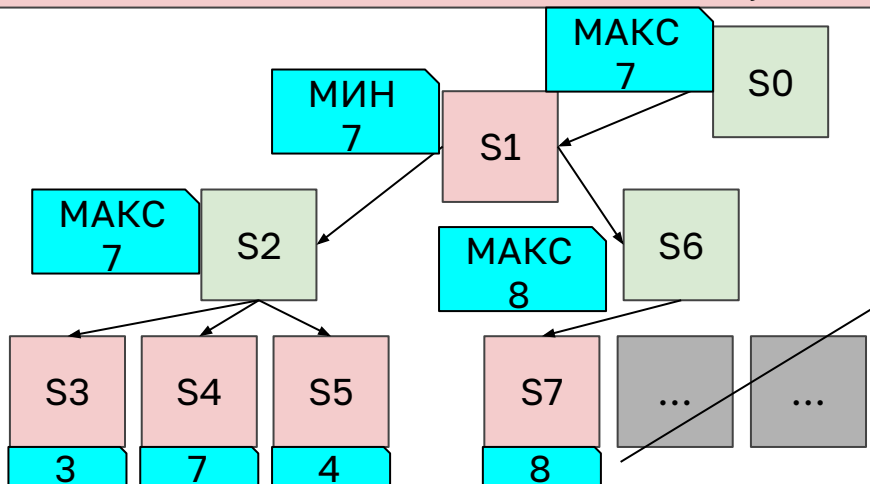
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



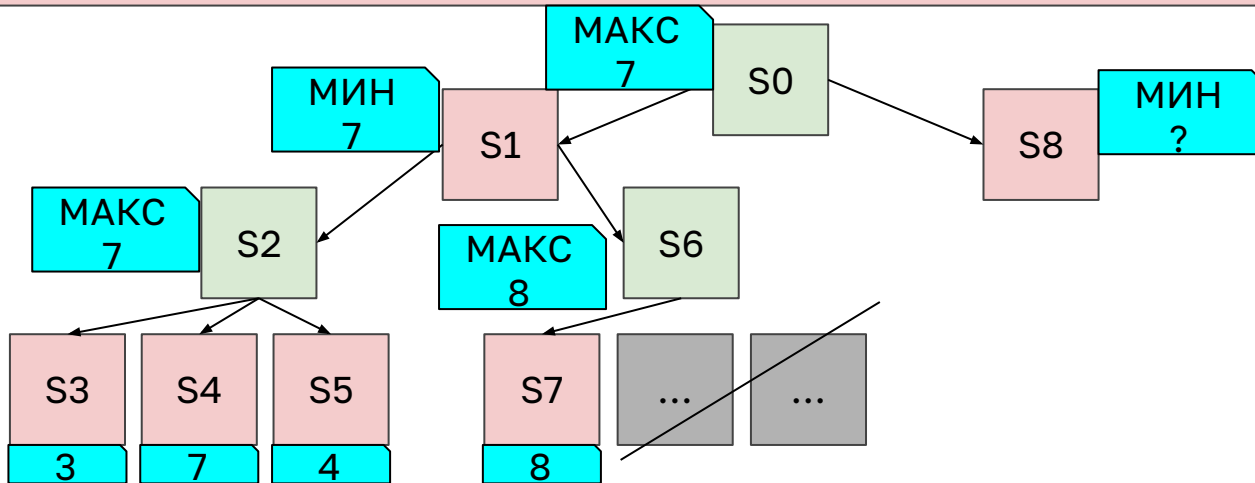
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



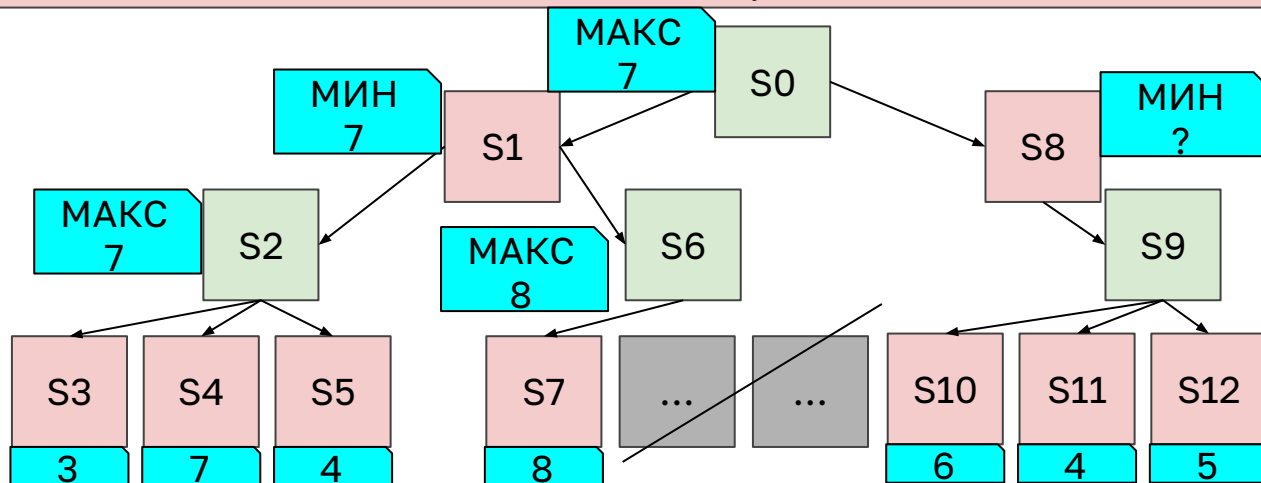
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



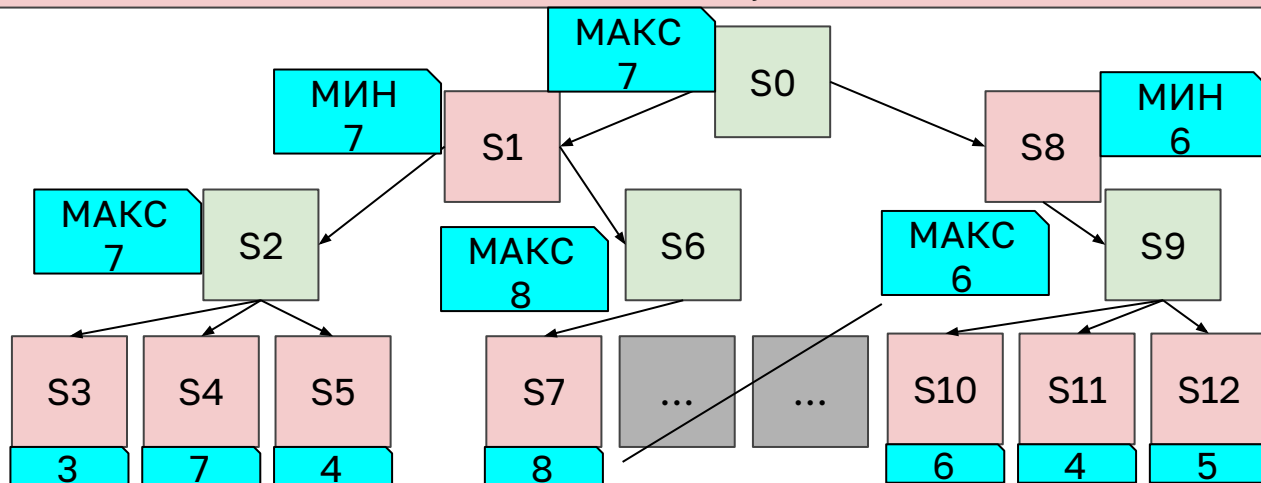
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



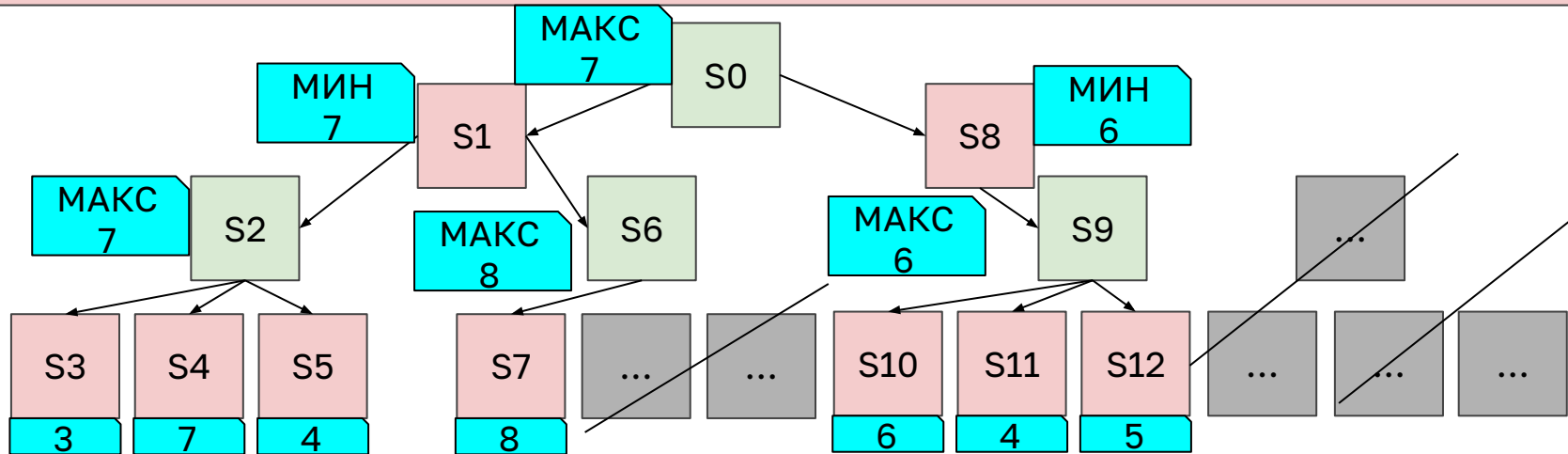
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



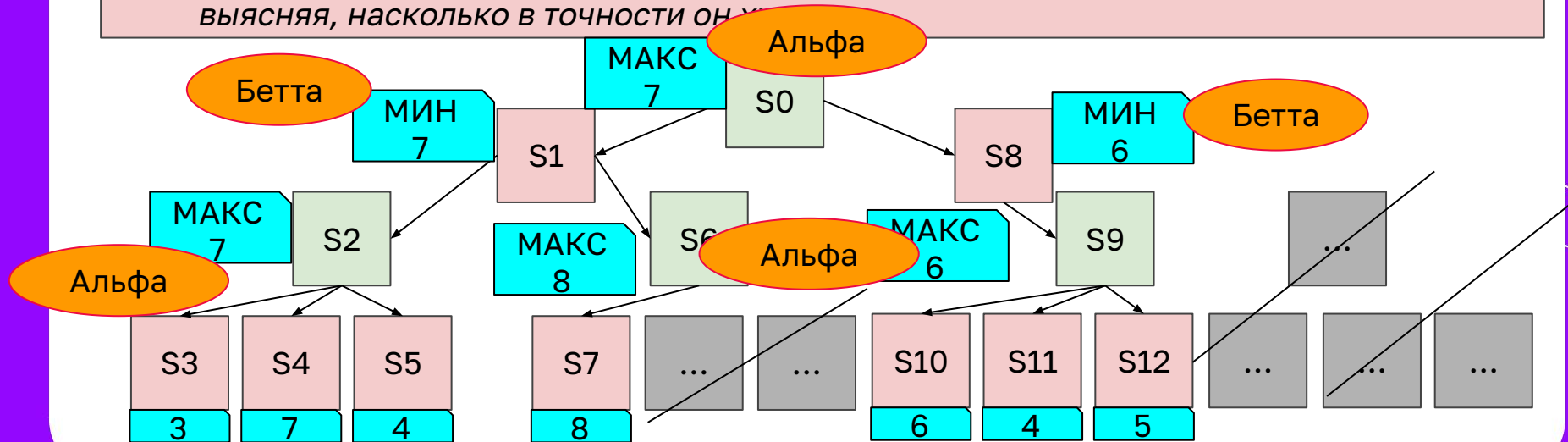
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



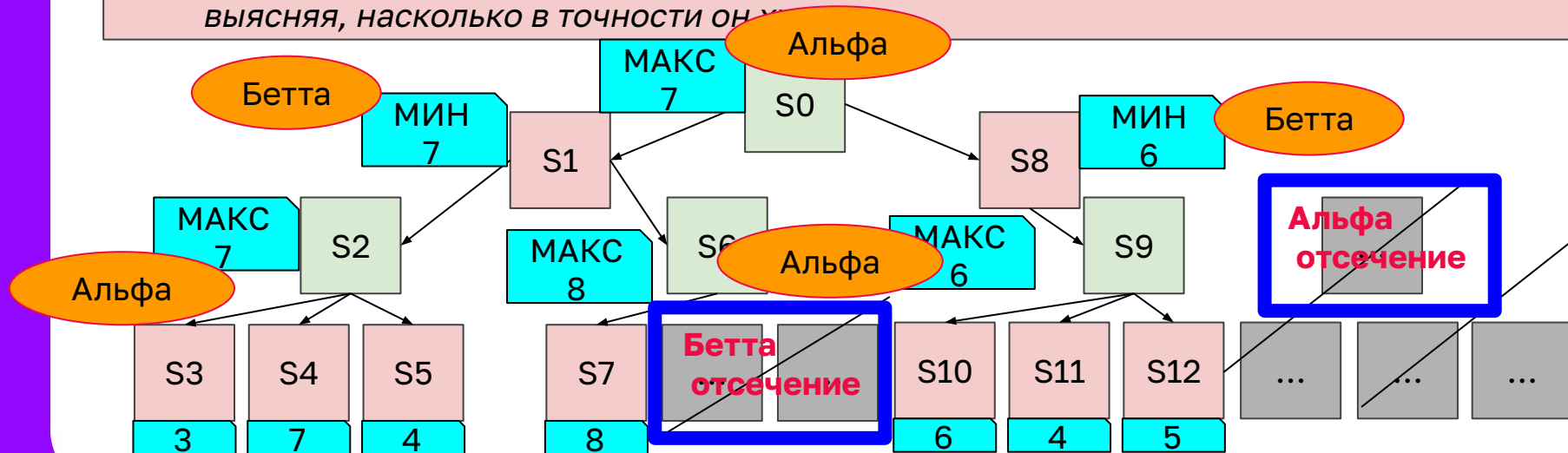
Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он худший.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он худший.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.

Общие правила:

- 1) концевая вершина игрового дерева оценивается статической оценочной функцией сразу, как только она построена;
- 2) промежуточная вершина предварительно оценивается по минимаксному принципу, как только стала известна оценка хотя бы одной из ее дочерних вершин; каждая предварительная оценка пересчитывается (уточняется) всякий раз, когда получена оценка еще одной дочерней вершины;
- 3) **следствие:** альфа-величины не могут уменьшаться, а бета-величины не могут увеличиваться.

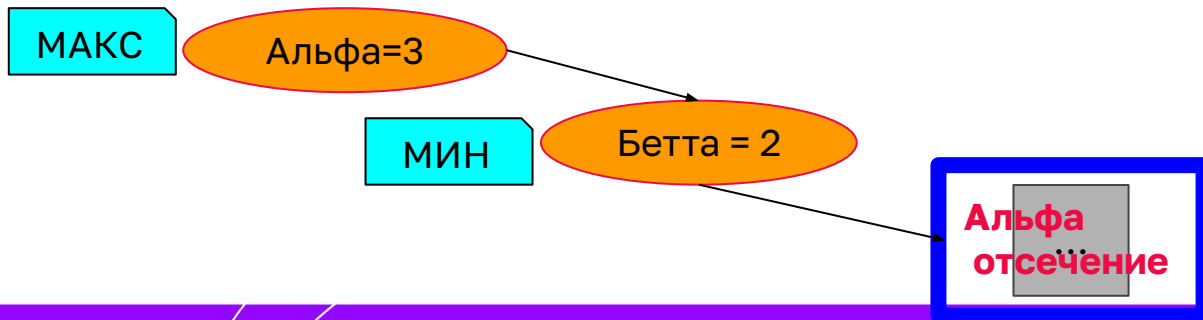
Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.

Правила Альфа и Бетта отсечений:

1. Альфа отсечение. Перебор можно прервать ниже, чем вершина с бета величиной, если **эта бета** величина \leq **альфа** величине **выше**.



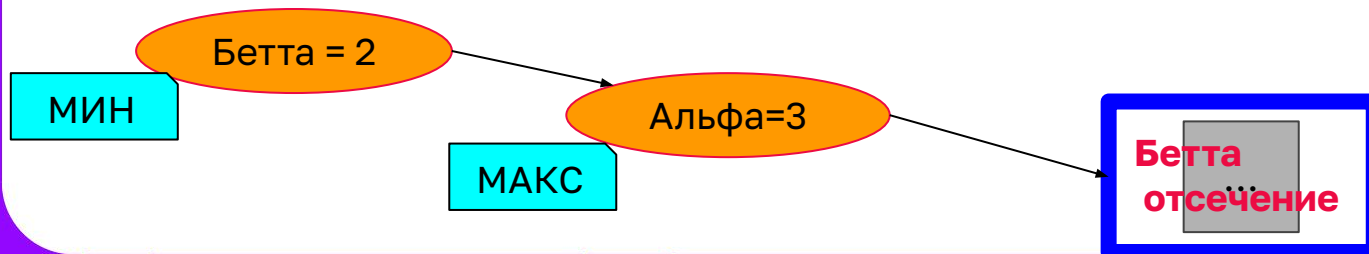
Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.

Правила Альфа и Бетта отсечений:

1. Альфа отсечение. Перебор можно прервать ниже, чем вершина с бетта величиной, если **эта бетта** величина \leq **альфа** величине **выше**.
2. Бетта Отсечение. Перебор можно прервать ниже, чем вершина с альфа величиной, если **эта альфа** величина \geq **бетта** величине **выше**.



Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)

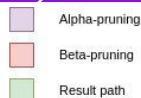


Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.

Правила Альфа и Бетта отсечений:

1. Альфа отсечение. Перебор можно прервать ниже, чем вершина с бетта величиной, если **эта бетта** величина \leq **альфа** величине **выше**.
2. Бетта Отсечение. Перебор можно прервать ниже, чем вершина с альфа величиной, если **эта альфа** величина \geq **бетта** величине **выше**.

“Выше” означает, что альфа/бетта величине **одной из предшествующих** альфа/бетта величин соответственно (включая корневую вершину дерева)



— ×

Например, в данном случае при проверке на отсечение в S15 мы смотрим как на S10, так и на S2. **По-другому**: мы можем считать, что нижестоящие альфа и бета значения как бы “наследуют” вышестоящие значения. Ну и помним про следствие: **альфа-величины не могут уменьшаться, а бета-величины не могут увеличиваться.**

90

Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.

Правила Альфа и Бетта отсечений:

1. Альфа отсечение. Перебор можно прервать ниже, чем вершина с бетта величиной, если **эта бетта** величина \leq **альфа** величине **выше**.
2. Бетта Отсечение. Перебор можно прервать ниже, чем вершина с альфа величиной, если **эта альфа** величина \geq **бетта** величине **выше**.

Утверждение: Альфа-бета процедура всегда приводит к тому же результату (наилучшему ходу), что и простая минимаксная процедура той же глубины.

Алгоритм МинМакс с Альфа-Бетта отсечениями (Альфа бетта процедура)



Суть: вычислять СОФ как только будет возможность, избегать лишних вычислений. В основе отсечений лежит достаточно очевидное соображение: если есть два варианта хода одного игрока, то **худший** в ряде случаев можно сразу **отбросить**, не выясняя, насколько в точности он хуже.

Правила Альфа и Бетта отсечений:

1. Альфа отсечение. Перебор можно прервать ниже, чем вершина с бетта величиной, если **эта бетта** величина \leq **альфа** величине **выше**.
2. Бетта Отсечение. Перебор можно прервать ниже, чем вершина с альфа величиной, если **эта альфа** величина \geq **бетта** величине **выше**.

Утверждение: Альфа-бета процедура всегда приводит к тому же результату (наилучшему ходу), что и простая минимаксная процедура той же глубины.

Результаты: статическая оценочная функция и альфа-бета процедура две неперенные составляющие почти всех компьютерных игровых программ

Примеры Статических оценочных функций (СОФ)



Игра “Крестики и нолики”:

1. 3X - это +10
2. 3O - это -10
3. $(\text{Cnt}(\text{Возможных победных строк } X) + \text{Cnt}(\text{Возможных победных столбцов } X) + \text{Cnt}(\text{Возможных победных диагоналей } X)) - (\text{Cnt}(\text{Возможных победных строк } O) + \text{Cnt}(\text{Возможных победных столбцов } O) + \text{Cnt}(\text{Возможных победных диагоналей } O))$

x ₃		
x ₁	o ₁	x ₂
o ₃		o ₂

$$(1+0+0) - (1+1+1) = -2$$

		X
	X	O

$$(2+2+2) - (1+1+0) = 4$$

Примеры Статических оценочных функций (СОФ)



Игра “Крестики и нолики”:

1. 3X - это +10
2. 3O - это -10
3. $(\text{Cnt}(\text{Возможных победных строк } X) + \text{Cnt}(\text{Возможных победных столбцов } X) + \text{Cnt}(\text{Возможных победных диагоналей } X)) - (\text{Cnt}(\text{Возможных победных строк } O) + \text{Cnt}(\text{Возможных победных столбцов } O) + \text{Cnt}(\text{Возможных победных диагоналей } O))$

Игра “Шашки”:

$\text{Cnt}(\text{свои}) - \text{Cnt}(\text{чужие}) + 10 * \text{cnt}(\text{свои дамки}) - 10 * \text{cnt}(\text{чужие дамки})$

Примеры Статических оценочных функций (СОФ)



Игра “Шахматы”:

Шеннон, Клод, 1950, "Программирование компьютера для игры в шахматы", Philosophical Magazine, Ser.7, Vol. 41, № 314.

“Он дает грубый пример функции оценки, в которой значения черных позиций вычитаются из значений белых. Материал оценивается по обычной относительной стоимости фигур (1 очко за пешку , 3 очка за коня или слона , 5 за ладью и 9 очков за ферзя или короля). Он учитывает такие факторы, как позиционные сдвоенные пешки , обратная пешка или изолированные пешки . Подвижность также является фактором, который добавляет 0,1 балла к каждому разрешенному движению. Он также считает мат взятием короля и дает ему искусственную ценность в 200 очков. ”

ГЛАВА 1 _ ПР1. Поиск по дереву

ПР1. Поиск по дереву (8 баллов)



Вариант 1. Закрепление материала.

Задание 1. Палочки. Произведите анализ игры “Палочки” со следующими правилами. Перед 2 игроками лежит 31 палочка на столе, за один ход игрок может взять 1,3 или 4 палочки. проигрывает тот, кто забирает последнюю палочку со стола. Какой игрок победит при правильной игре? (1 балл)

Задание 2. Альфа-бета отсечения. Придумайте некоторое дерево игры, убедитесь на примере в утверждении “Альфа-бета процедура всегда приводит к тому же результату (наилучшему ходу), что и простая минимаксная процедура той же глубины”. Для этого: итеративно проиграйте это дерево с помощью альфа-бетта отсечения. Затем: проверьте результат минимаксной процедурой на полном дереве. (3 балла)

Задание 3. СОФ. 1) Выберите и проанализируйте игру; 2) Предложите СОФ для этой игры; 3) Возьмите несколько (3-4) ситуаций игры и покажите насколько адекватна и применима предложенная СОФ.

Игры: “Шашки”, “Пять в ряд”, “Лиса и гуси”, “Бридж Ит”, “Рассада”, “Так-тиклъ”, “ЦЗЯНЬШИДЗЫ”, “Шестнадцать солдат”, “Пентамино”, “Щелк” (4 балла)

ГЛАВА 1 _ ПР1. Поиск по дереву

ПР1. Поиск по дереву (8 баллов)



Вариант 2. Для тру прогеров. (+3 доп. балла)

Выбрать игру и реализовать ее на языке высокого уровня с интерфейсом с ИИ на основе **альфа-бетта отсечений**.

PS. Отличная практика Алгоритмов и структур данных!

Например, игры: “Шашки”, “Пять в ряд”, “Лиса и гуси”, “Бридж Ит”, “Рассада”, “Так-тикль”, “ЦЗЯНЬШИДЗЫ”, “Шестнадцать солдат”, “Пентамино”, “Щелк”

**Спасибо
за внимание!**

itMO *re than a*
UNIVERSITY

tatiana.atyapsheva@mail.ru
abrosimov.kirill.1999@mail.ru