ივანე ჯავახიშვილის სახელობის თბილისის
სახელმწიფო უნივერსიტეტი

ზუსტ და საბუნებისმეტყველო მეცნიერებათა ფაკულტეტი

კომპიუტერულ მეცნიერებათა დეპარტამენტი

გრიგორ ელოიანი

სემინარი თემაზე:

საჭადრაკო კომპიუტერული პროგრამების მუშაობის ალგორითმები და პრინციპები

ხელმძღვანელი: მანანა ხაჩიძე,

სრული პროფესორი

თბილისი, 2015

სარჩევი

ანოტაცია

ჭადრაკის სრულყოფილი მოთამაშის შექმნის მცდელობა ადამიანებს ჯერ კიდევ კომპიუტერის შექმნამდე ჰქონდათ, თუმცა რეალურად ამის შესაძლებლობა გაჩნდა გასული საუკუნის 50-ან წლებში. კომპიუტერის გამოგონებისთანავე დაიწყო მუშაობა საჭადრაკო პროგრამების შექმნაზე. თავდაპირველად ამას მხოლოდ ექსპერემენტული მიზანი ჰქონდა, ვინაიდან არავის სჯეროდა რომ კომპიუტერს შეეძლო უძლიერესი მოჭადრაკეების დამარცხება. მაგრამ როგორც აპარატურული შესაძლებლობების პროგრესმა, ასევე ახალი ალგორითმების და ევრისტიკების გამოყენებამ განაპირობა ის, რომ ბოლო 30 წლის განმავლობაში საჭადრაკო კომპიუტერულმა პროგრამებმა საგრნობი პროგრესი განიცადეს. ამ სემინარში განხილულია ასეთი პროგრამების მუშაობის პრინციპები, მათში გამოყენებული ალგორითმები და სხვადასხვა გაუმჯობესების მეთოდები.

საჭადრაკო პროგრამების განვითარების ისტორია.

ადამიანი ცდილობდა საჭადრაკო ავტომატის შექმნას ჯერ კიდევ მაშინ, როცა არ არსებობდა კომპიუტერი. ერთ ერთი პირველი, ვინც შეიმუშავა მანქანისთვის ჭადრაკის თამაშის ალგორითმი, იყო ცნობილი ბრიტანელი მათემატიკოსი ალან ტიურინგი. მან 1947 წელს პირველმა აღწერა ასეთი ალგორითმი.

დაახლოებით იმავე პერიოდში ამ ამოცანაზე კიდევ ერთი ცნობილი მათემატიკოსი, კლოდ შენონიც მუშაობდა, მან 1949-1950 წლებში გამოკვეთა ერთ ერთი მთავარი პრობლემა - ყოველ მომდევნო სვლაზე შესაძლო სვლების რაოდენობა იზრდება. გამოიყო ვარიანტების გადარჩევის ორი სტრატეგია. პირველი გულისხმობს ყველა შესაძლო ვარიანტის გადარჩევას, ხოლო მეორე - „ცუდი“ სვლების ამოგდებას მოჭადრაკეების გამოცდილებიდან გამომდინარე.

საჭადრაკო პროგრამების პრინციპები და ალგორითმები.

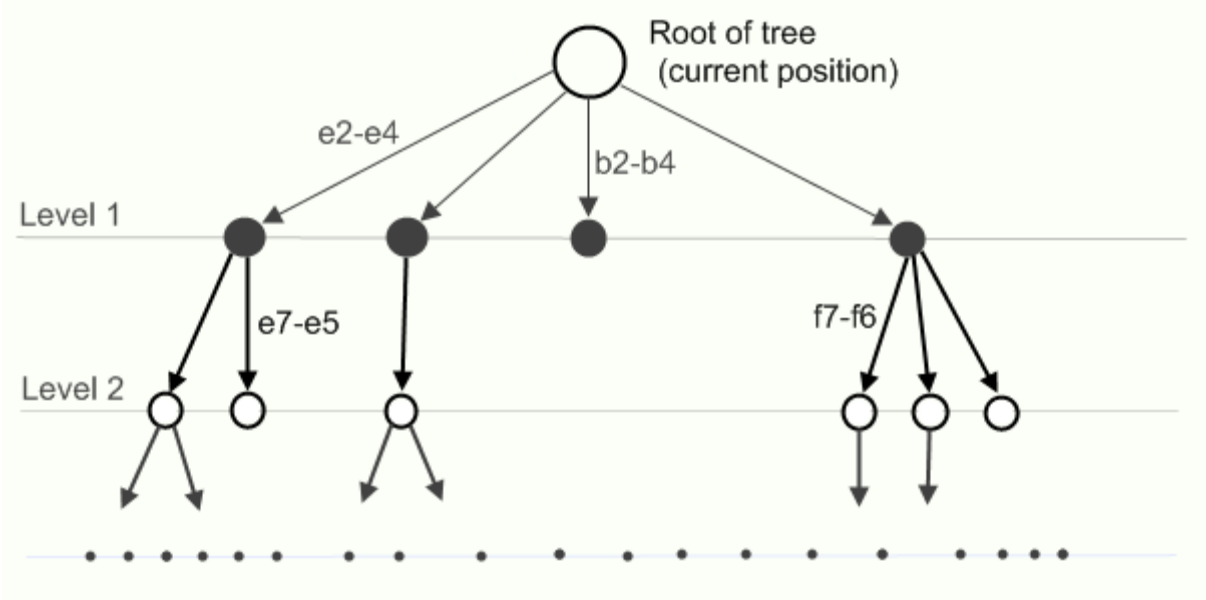
როგორც უკვე ითქვა, მთავარი პრობლემაა ვარიანტების ძალიან დიდი რაოდენობა. ჩვეულებრივ პოზიციაში საშუალოდ არსებობს დაახლოებით 40 შესაძლო სვლა ერთი მოთამაშისთვის და დაახლოებით ამდენივე მეორესთვის. ანუ ერთი სვლის სიღრმეზე შესაძლო ვარიანტების რაოდენობა უდრის 40*40=1600. ორი სვლის სიღრმეზე 2,5 მილიონი, ხოლო სამზე - 4 მილიარდი! მათემატიკოსების შეფასებით, ჭადრაკის პარტიის ყველა შესაძლო სვლების რაოდენობა დაახლოებით უდრის 10^40. თუ კომპიუტერის მონაცემთა ბაზაში შევიტანთ აბსოლუტურად ყველა შესაძლო პოზიციას, მისი თამაში იქნება იდეალური ნებისმიერი პოზიციიდან. მაგრამ რა თქმა უნდა ამხელა ინფორმაციას ვერც ერთი კომპიუტერი ვერ დაიტევს. შესაბამისად კომპიუტერული პროგრამებიც გარკვეულწილად ადამიანის სტრატეგიით მოქმედებენ - ითვლიან უახლოეს შესაძლო სვლებს, აფასებენ პოზიციას და ირჩევენ მისთვის საუკეთესო სვლას. მაგრამ როგორ აკეთებენ ამას?

მინიმაქსის ალგორითმი.

ისევე როგორც ყველა თამაშში, სადაც ორი მოთამაშე ერთმანეთის დამარცხებას ცდილობს, ჭადრაკშიც მინიმაქსის ალგორითმი საბაზისოა. ეს არის ალგორითმი, სადაც იგება ხე, რომლის კვანძები წარმოადგენს ყველა შესაძლო სვლას გარკვეულ პოზიციაში. ყოველ სვლას შეესაბამება გარკვეული შეფასება (რიცხვი, დადებითი, თუ

პოზიცია უკეთესია თეთრებისთვის, უარყოფითი, თუ უკეთესია შავებისთვის). ხის ყოველ დონეზე ხდება მონაცვლეობით იმ კვანძის ძებნა, რომელიც საუკეთესო შედეგს მოუტანს პირველ მოთამაშეს (მაქსი), ხოლო მომდევნოზე, რომელიც საუკეთესო შედეგს მოუტანს მეორე მოთამაშეს (მინი).

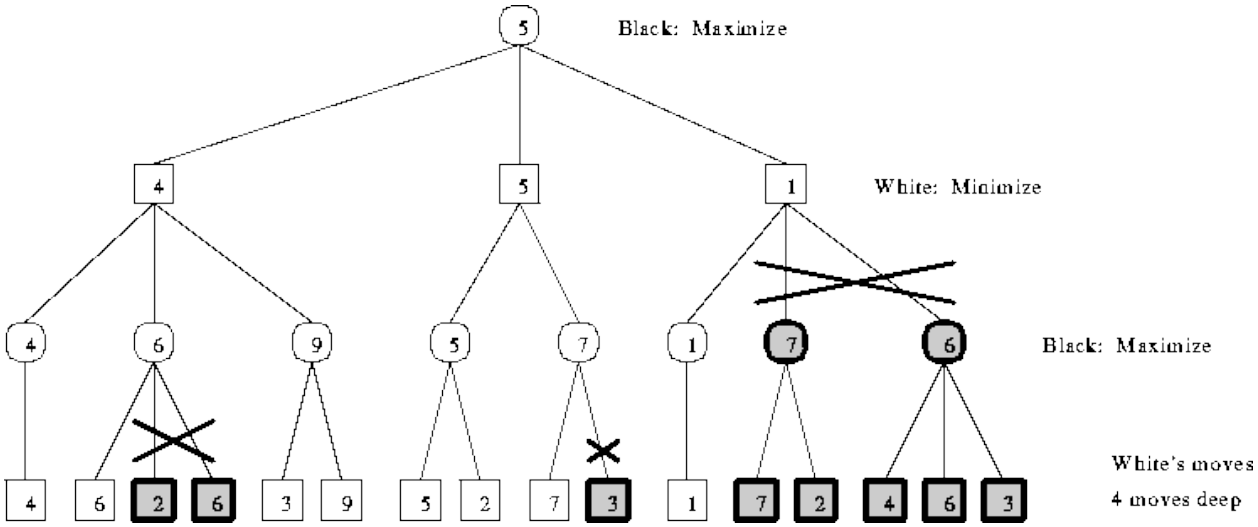
ეს სქემა პრაქტიკულად შეესაბამება იმას, თუ რა სტრატეგიით ფიქრობს ადამიანი ჭადრაკის თამაშისას. ანუ მოთამაშე თავის სვლებში ცდილობს იპოვოს საუკეთესო მისთვის, შემდეგ იგი ვარაუდობს, რომ მისი მოწინააღმდეგე ეცდება უკვე თავისთვის საუკეთესო სვლის მოძებნას, ამიტომ პირველი მოთამაშე ჯერ მოახდენს თავისი პოზიციის მაქსიმიზაციას, შემდეგ მოწინააღმდეგის პოზიციის მაქსიმიზაციას, ანუ თავისი პოზიციის მინიმიზაციას, შემდეგ ისევ საკუთარი პოზიციის მაქსიმიზაციას და ა.შ.



რა თქმა უნდა, იდეალურ შემთხვევაში, კომპიუტერი შეძლებდა ყველა შესაძლო სვლებითვის ასეთი ხის აგებას და მისთვის ყოველთვის იქნებოდა ცნობილი, თუ რომელი სვლა უნდა გაეკეთებინა გამარჯვების მოსაპოვებლად, მაგრამ როგორც უკვე აღინიშნა, ყველა შესაძლო ვარიანტების რაოდენობა კომპიუტერისთვისაც კი ძალიან დიდია. ამიტომაც მინიმაქსის ალგორითმის გამოყენებისას, პროგრამა ითვლის სვლებს მხოლოდ ხის გარკვეულ N სიღრმემდე, რომელიც განსხვავებული იქნება პარტიის დროიდან და პოზიციის სირთულიდან გამომდინარე. და მაინც, რამხელა ხეს აგებს პროგრამა ვარიანტების დასათვლელად? როგორც ზემოთ იყო ახსნილი, იმისთვის რომ 3 სვლის სიღრმეზე მოხდეს ყველა შესაძლო ვარიანტის განხილვა, ასეთი ვარიანტების რაოდენობა იქნება დაახლოებით 4 მილიარდის ტოლი, რაც უკვე საკმაოდ დიდი რიცხვია კომპიუტერისთვის. მაგრამ როგორც ცნობილია, იმისთვის რომ შეძლო ჭადრაკის თამაში მაღალ დონეზე, სამი სვლის გამოთვლა ძალიან ცოტაა. ამიტომაც მხოლოდ მინიმაქსის ალგორითმის გამოყენება ყველა ვარიანტისთვის არ იქნებოდა სწორი. გაჩნდა ახალი მეთოდების გამოყენების საჭიროება. ერთ-ერთი ასეთი მეთოდია ალფა-ბეტა კვეთა.

ალფა-ბეტა კვეთა

როგორ ცდილობს ადამიანი მისთვის საუკეთესო სვლის პოვნას, როდესაც ახდენს ვარიანტების გადარჩევას? რა თქმა უნდა, არსებობენ პირველივე შეხედვით იმდენად ცუდი ვარიანტები, რომ მათ განხილვას ადამიანი არც კი აგრძელებს. მაგალითად თუ მოთამაშე გამოთვლის, რომ სვლების გარკვეული მიმდევრობის შემდეგ, მისი მოწინააღმდეგე კლავს ლაზიერს, იგი აღარ გააგრძელებს ამ ვარიანტის განხილვას, ვინაიდან გასაგებია რომ ეს შედეგი მისთვის ძალიან ცუდია. ანუ მათემატიკურად რომ ვთქვათ, მინიმაქსის ხეში მოიძებნება ისეთი კვანძები რომელთა ქვეხეში ყველა კვანძის შეფასება იქნება იმაზე ნაკლები, ვიდრე რომელიმე უკვე ნაპოვნი კვანძის შეფასება. გამოდის რომ შეგვიძლია მათემატიკური კორექტირების შენარჩუნებით ხის გარკვეული ქვენაწილი მოვკვეთოთ ისე, რომ საბოლოო შეფასებაზე ეს გავლენას არ მოახდენს. სწორედ ამას აკეთებს ალფა-ბეტას ალგორითმი. ეს ალგორითმი პირველად 1958 წელს გამოიყენეს. მისი დახმარებით სვლების შესაძლო ვარიანტების რაოდენობა საგრძნობლად შემცირდა, რამაც მისცა საშუალება კომპიუტერებს გაცილებით დიდ სიღრმეზე დაეთვალათ ვარიანტები. თუმცა იმ დროისთვის კომპიუტერები ჯერ კიდევ ვერ უწევდნენ სერიოზულ წინააღმდეგობას მსოფლიოს მოწინავე მოჭადრაკეებს.



თეორიის გამოყენება პრაქტიკაში

ჩვენ უკვე ვიცით, რომ პარტიაში ვარიანტების დასათვლელად კომპიუტერი იყენებს მინიმაქსის და ალფა-ბეტას ალგორითმებს. მაგრამ არაფერი გვითქვავს

ჭადრაკის მთავარ თავსათეხზე - როგორ ხვდება კომპიუტერი, სხვადასხვა პოზიციებში რომელია უკეთესი პოზიცია და რომელი უარესი?

საჭადრაკო ძრავებში არსებობს ე.წ. პოზიციის შეფასების ფუნქცია, რომელიც გამოიძახება ყველა იმ პოზიციისთვის, რომელსაც განიხილავს პროგრამა, შემდეგ ეს შეფასება (ქულა) ენიჭება მინიმაქსის ხეში პოზიციის შესაბამის კვანძს. ასე ვიღებთ აწყობილ ხეს, რომელშიც ვარიანტების გადარჩევაც უკვე განვიხილეთ ზემოთ. სინამდვილეში შეფასების ფუნქციის მუშაობის დრო არის ყველაზე მნიშვნელოვანი დროის მაჩვენებელი საჭადრაკო ძრავებში, ამიტომ არის მნიშვნელოვანი მინიმაქსის ალგორითმში ალფა-ბეტა კვეთის გამოყენება, ვინაიდან რაც ნაკლები პოზიციისთვის (ანუ კვანძისთვის) იქნება დასათვლელი შეფასება, მით ნაკლები დრო დაიხარჯება მთლიანობაში.

კი მაგრამ, როგორ არის მოწყობილი შეფასების ფუნქცია? სწორად მასზეა დამოკიდებული, როგორ აფასებს პროგრამა ამა თუ იმ პოზიციას და შესაბამისად ისიც, თუ რა სვლას აირჩევს, ეს კი თავის მხრივ განსაზღვრავს პარტიის შედეგს.

მაღალი დონის მოჭადრაკემ ყოველთვის იცის, თუ რატომ ირჩევს ამა თუ იმ სვლას, მისთვის ადვილი დასანახია მატერიალური თუ პოზიციური უპირატესობა. მაგრამ ჭადრაკი იმდენად კომპლექსური თამაშია, რომ არ არსებობს ზუსტი წესების ჩამონათვალი, რომელიც მოგვცემს იმის თქმის საშუალებას, რომ ამ წესების დაცვით ჩვენ ყოველთვის უკეთესი პოზიცია გვექნება, რომელიც საბოლოო გამარჯვებამდე მიგვიყვანს.

ამიტომაც შეფასების ფუნქციაში გამოყენებულია ის თეორიული ცოდნა, რაც წლების მანძილზე ჭადრაკის თამაშის და მისი კვლევის შედეგად იქნა დაგროვებული. ასე მაგალითად, პირველ რიგში ფუნქცია ითვლის დაფაზე არსებული ფიგურების შეფასებებს, როგორც დიდი ხნის პრაქტიკამ აჩვენა, პაიკი ფასდება პირობითად 1 ქულით, კუ და მხედარი - 3 ქულით, ეტლი 5 ქულით და ლაზიერი 9 ქულით. მაგრამ რა თქმა უნდა მხოლოდ ფიგურების რაოდენობა არ არის საკმარისი იმისთვის, რომ განისაზღვროს, თუ რომელ მოთამაშეს აქვს უპირატესობა, ვინაიდან შეიძლება ერთ მოთამაშეს უფრო მეტი ფიგურა ჰქონდეს დარჩენილი, მაგრამ იმდენად ცუდი პოზიციური განლაგება ჰქონდეს, რომ სულ რამდენიმე სვლაში გარდაუვალი შამათი ელოდება. ამიტომაც შეფასების ფუნქცია იყენებს უამრავ კრიტერიუმს პოზიციის უფრო ზუსტი, უფრო რეალური შეფასებისთვის. როგორც უკვე ითქვა, არ არსებობს ზუსტი წესების და კრიტერიუმების ჩამონათვალი, რომელიც იძლევა საშუალებას ითქვას, თუ რა არის ჭადრაკში ფიგურების კარგი განლაგება და რა - ცუდი. ამიტომაა, რომ სხვადასხვა საჭადრაკო პროგრამები სხვადასხვანაირად აფასებენ ერთსა და იმავე პოზიციას (რა თქმა უნდა დღეისთვის უძლიერესი პროგრამები დაახლოებით ერთნაირად აფასებენ ერთსა და იმავე პოზიციას, ვინაიდან ყველა მათგანი დაიხვეწა უახლოესი წლების გამოცდილებიდან გამომდინარე).

მოვიყვანოთ პოზიციის შეფასების რამდენიმე ფაქტორი და მათი მიახლოებითი წონები:

- ორი კუს უპირატესობა: +0,3 ქულა
- მეფე დაფარულია საკუთარი პაიკებით: +0,5
- მხედარი ან ლაზიერი იმყოფება საკუთარი ან მოწინააღმდეგე მეფის სიახლოვეს: +0,01
- სუსტი ან გაორებული პაიკ(ებ)ი: -0,25
- ეტლი ნახევრადღია ვერტიკალზე: +0,05
- ეტლი ღია ვერტიკალზე: +0,1
- ორი ეტლი მე-7 ჰორიზონტალზე: +0,5

და ა.შ.

ასეთი ფაქტორების რაოდენობა ძალიან დიდია, გარდა ამისა, ბევრი ფაქტორის წონა იცვლება დაფაზე არსებული სიტუაციიდან გამომდინარე, მაგალითად ლაზიერების გაცვლამდე მეფის დაცულობა უფრო მაღალი წონით აღინიშნება, ვიდრე გაცვლის შემდეგ.

ერთი შეხედვით ყველაფერი მარტივადაა, ვითვლით მოთამაშის ფიგურების ჯამს, ვუმატებთ მისი პოზიციის ფაქტორების წონების ჯამს, ვაკეთებთ იგივეს მისი მოწინააღმდეგისთვის და მორჩა, ჩვენ უკვე ვიცით როგორ ვითამაშოთ.

მაგრამ ყველაფერი ასეთი მარტივი არაა. პირველი პროგრამები მუშაობდნენ ზემოთ აღწერილი ალგორითმით - არჩევდნენ ვარიანტებს ფიქსირებულ სიღრმეზე შეფასების ფუნქციის გამოძახებით. მაშინვე გაირკვა, რომ გადარჩევა შესაძლოა შეწყდეს ფიგურების შუა გაცვლაში, ან მაშინ, როცა ერთ სვლაში შესაძლოა შამათი მივიღოთ, ეს კი ნიშნავს, რომ შეფასების ფუნქიის შედეგს მნიშვნელობა აღარ ექნება. იყო მცდლობები შეფასების ფუნქციაში ამ ყველაფრის გათვალისწინების, მაგრამ იგი ძალიან ნელი და რთული ხდებოდა. გარდა ამისა ჩნდებოდა ისეთი ფაქტორი, როგორიც არის ე.წ. „ჰორიზონტის ეფექტი“, როდესაც პროგრამა ცდილობს მისთვის არასასურველი შედეგის რამდენიმე სვლით გადაწევას პოზიციის გაუარესების ხარჯზე, ანუ მოვლენათა ჰორიზონტიდან გააქროს უსიამოვნებები.

სელექციური მეთოდები ფაქტიურად ჩავარდა. შეუძლებელი იყო იმის გარკვევა, თუ როგორ თამაშობს ადამიანი ჭადრაკს, ხოლო ქვეხის კვეთა მისი განხილვის გარეშე საჭადრაკო პრინციპებიდან გამომდინარე, არ გამოდგა სწორი ტაქტიკა. ანუ ყოველთვის შეიძლება მივიღოთ ისეთი სვლა, როდესაც პროგრამა გამორიცხავს საჭირო სვლას.

ბოლოს და ბოლოს, გაირკვა, რომ შეფასების ფუნქციის გამოძახება შეიძლება მხოლოდ „მშვიდ“ პოზიციებში. ამიტომ ჩვეულებრივი გადარჩევის ბოლოს, დაემატა კიდევ ერთი, გამარტივებული გადარჩევა, სადაც განიხილება მხოლოდ ფიგურის აყვანის, პაიკის გარდაქმნის, ქიშის და ქიშზე პასუხის სვლები. ვინაიდან, ასეთი სვლების რაოდენობა არც ისე დიდია, საბოლოო სიჩქარე საგრძნობლად არ ნელდება, სამაგიეროდ პროგრამის საიმედოობა მკვეთრად იზრდება.

გაუმჯობესების მეთოდები

გასული საუკუნის 70-ანი წლებიდან ხდებოდა საჭადრაკო პროგრამების თამაშის ხარისხის საგრძნობი გაუმჯობესება. მთელი ამ ხნის მანძილზე ვერ მოხდა ვერც ერთი საბაზისო ალგორითმის მოფიქრება, რომელიც ალფა-ბეტას შეცვლიდა. პროგრამების გაუმჯობესება ხდებოდა მათ შორის აპარატურული მონაცემების გაუმჯობესების ხარჯზე. გაჩნდნენ ე.წ. სუპერკომპიუტერები. პირველი ასეთი კომპიუტერი შეიქმნა Bell Laboratories-ში კენ ტომპსონის მიერ, და მას შეეძლო წამში 180000 ოპერაციის შესრულება. შედარებისთვის, იმ დროისთვის ჩვეულებრივ კომპიუტერს შეეძლო მხოლოდ 5000 ოპერაციის შესრულება წამში. ამ კომპიუტერს შეეძლო 8-9 სვლის სიღრმეზე პოზიციის გაანალიზება, რაც ჭადრაკში ოსტატის წოდების დონეა. ამ კომპიუტერმა იმარჯვა საერთაშორისო საჭადრაკო ტურნირში კომპიუტერებს შორის და 80-იანი წლების დასაწყისში სხვა რამდენიმე ტურნირში.

მაგრამ საჭადრაკო პროგრამების გაუმჯობესება ხდებოდა არა მხოლოდ აპარატურის გაუმჯობესების ხარჯზე. მართალია, ალგორითმიების კარდინალური ცვლილებები არ მომხდარა, მაგრამ გაჩნდა უამრავი სასარგებლო გაუმჯობესება. მიმოვიხილოთ რამდენიმე მათგანი:

- დებიუტების ბაზის გამოყენება - დებიუტების თეორია კარგად არის შემუშავებული და ცხადია, პროგრამებმაც დაიწყეს მისი გამოყენება, თანაც ადამიანზე უკეთესად, ვინაიდან რამდენიმე მილიონი პოზიციის დამახსოვრება კომპიუტერისთვის არანაირ პრობლემას არ წარმოადგენს.
- მთლიანად გათვლილი ენდშპილების ბაზის გამოყენება. 80-ანი წლებიდან კენ ტომპსონმა დაიწყო 4 ფიგურიანი ენდშპილების ბაზის შემუშავება. დღეისთვის კი არსებობს უკვე 6 ფიგურიანი (მეფის ჩათვლით) ენდშპილის ბაზები. ამ ბაზების გამოყენებით პროგრამები მარტვი ენდშპილებს სრულყოფილი სიზუსტით თამაშობენ. ამიტომაც კომპიუტერის წინააღმდეგ თამაშისას უპირატესობის მოპოვება ენდშპილში ფაქტიურად შეუძლებელია.
- სვლების გამოთვლა მაშინაც კი, როცა მოწინააღმდეგის სვლის ჯერია. ისევე როგორც ადამიანი, კომპიუტერიც არ კარგავს დროს და მოწინააღმდეგის სვლაზეც აგრძელებს ვარიანტების დათვლას.
- სხვადასხვა ევრისტიკების გამოყენება, რომლებიც ასორტირებენ სვლებს გარჩევის პროცესში. ალფა ბეტა ალგორითმისთვის საკმაოდ მნიშვნელოვანი ფაქტორია სვლების მიმდევრობა, რომლებსაც იგი განიხილავს. თუ მიმდევრობა სწორად არის დალაგებული, ალგორითმი მნიშვნელოვნად ჩქარდება. ვინაიდან ალგორითმი ექსპონენციალურია, N+1 სვლის სიღრმეზე იგი მუშაობს გაცილებით ნელა, ვიდრე N სიღრმეზე. თუ დაულაგებელი სვლების მიმდევრობისთვის საშუალოდ უნდა განიხილებოდეს 40^N სვლა, დალაგებული მიმდევრობისთვის, როგორც აჩვენა პრაქტიკამ, საკმარისია დაახლოებით 6^N სვლის გადარჩევა. მაგალითად ერთ-ერთი ასეთი

ევრისტიკაა, რომ ჯერ განიხილება სვლა, სადაც ხდება ფიგურის აყვანა, ვინაიდან ხშირ შემთხვევაში ეს სვლა საუკეთესო ან ერთ ერთი საუკეთესოა.

- იტერატიული ჩაღრმავება (iterative deepening) - რეალურ თამაშში პროგრამას გააჩნია გარკვეული დრო ყოველ სვლაზე, და ამ დროის სწორად გამოყენებაა საჭირო. უბრალოდ გადარჩვა ფიქსირებულ სიღრმეზე არასწორია - რთულ პოზიციაში შესაძლო ვარიანტების რაოდენობა დიდია, და მათში ჩაღრმავებას პროგრამა დიდ დროს მოანდომებს და შესაძლოა ვერ მოასწროს გადარჩევა. ხოლო ალფა-ბეტას ერთ-ერთი თვისებაა, რომ არ შეიძლება დაუსრულებელი გადარჩევის შედეგის გამოყენება. მეორეს მხრის, ჩაკეტილ პოზიციებში, სადაც ვარიანტების რაოდენობა მცირეა, გადარჩევა შესაძლოა საკმაოდ მალე დასრულდეს. იტერატიული ჩაღრმავების იდეა იმაშია, რომ პროგრამა აკეთებს გადარჩევას N სვლის სიღრმეზე, შემდეგ თუ დარჩა დრო, N + 1 სიღრმეზე, შემდეგ თუ ისევ დარჩა დრო, N + 2 და ა.შ. ვინაიდან შემდეგ დონეზე გადარჩევა საშუალოდ მოითხოვს 6-ჯერ დიდ დროს, შესაძლოა პროგრმამ ვეღარ მოასწროს გადარჩევის დასრულება, და ამ შემთხვევაში იგი გამოიყენებს წინა იტერაციიდან მიღებულ შედეგებს.
- ჰეშ-ცხრილები - ვინაიდან ჭადრაკში ერთი და იგივე პოზიციის მიღება შესაძლებელია სვლების სხვადასხვა მიმდევრობით, გაჩნდა იდეა, რომ პროგრამას შეენახა უკვე გადარჩეული პოზიციის შესახებ ინფორმაცია (შეფასება, გადარჩევის სიღრმე და საუკეთესო სვლა). ასეთ შემთხვევაში, თუ ეს პოზიცია განმეორდება, პროგრამა აღარ დახარჯავს დროს ამ პოზიციის გადარჩევაზე, ან გააგრძელებს მის გადარჩევას უფრო დიდ სიღრმეზე, ვიდრე მანამდე ჰქონდა გარჩეული.

ადამიანი კომპიუტერის წინააღმდეგ

პირველი კომპიუტერის შექმიდან მცირე ხანშივე გაჩნდა საჭადრაკო პროგრამის შექმნის იდეა. პირველი ექსპერიმენტული პროგრამა შეიქმნა 1950 წელს კომპიუტერზე, რომელსაც შეეძლო წამში 10000 ოპერაციის შესრულება. რამდენიმე წელიწადში კი კომპიუტერულმა პროგრამამ დამწყები მოჭადრაკე 23 სვლაში დაამარცხა, თუმცა ძლიერ მოჭადრაკესთან მარცხი განიცადა. საჭადრაკო პროგრამების პროგრესი არც ისე შასამჩნევი იყო მე-20 საუკუნის 80-ან წლებამდე, როდესაც დაიწყო სუპერკომპიუტერების ერა, რომლებსაც შეეძლოთ ასი ათასობით ოპერაციის შესრულება წამში. 80-იანი წლების დასაწყისში ყველაზე მძლავრი კომპიუტერი, რომელზეც საჭადრაკო პროგრამა შეიქმნა, იყო Cray X_MP. შემდეგ გაჩნდნენ ისეთი პროგრამები, როგორებიცაა Chip Test და Deep Thought, რომლებსაც წამში ნახევარ მილიონამდე ოპერაციის შესრულება შეეძლოთ. Deep Thought-მა 80-ანი

წლების ბოლოს შეძლო რამდენიმე დიდოსტატის დამარცხება. ამ ფაქტმა დიდი ინტერესი გამოიწვია კომპანია IBM-ში, რომლებმაც გააგრძელეს საჭადრაკო მანქანის შექმნაზე მუშაობა. ასე შეიქმნა კომპიუტერი Deep Blue, რომელიც სპეციალურად ჭადრაკის თამაშისთვის შეიქმნა. იგი შედგებოდა 31 ძირითადი პროცესორისგან, და ყოველ ძირითად პროცესორს 16 სპეციალიზირებული საჭადრაკო პროცესორი ჰქონდა, საერთო ჯამში კი Deep Blue -ს შეეძლო წამში 100 მილიონი პოზიციის შეფასება! დადგა საჭადრაკო პროგრამების ისტორიაში ყველაზე მნიშვნელოვანი მატჩის დრო - მსოფლიოს უძლიერესი კომპიუტერი მსოფლიო ჩემპიონის წინააღმდეგ. ასეთი მატჩი შედგა 1996 წელს და მაშინ მსოფლიო ჩემპიონმა გარი კასპაროვმა შეძლო Deep Blue-ს დამარცხება ანგარიშით 4:2. თუმცა ამით არ დამთავრებულა Deep Blue-ი ისტორია. IBM-მა გაითვალისწინა პროგამის სისუსტეები, ასევე გაზარდა სიმძლავრე წამში 200 მილიონ პოზიციამდე და 1997 შეგდა ე.წ. მატჩ-რევანში, სადაც პირველად ისტორიაში კომპიუტერმა მოუგო მატჩი მსოფლიო ჩემპიონს. შეიძლება ითქვას, რომ ამ მომენტიდან მოყოლებული კომპიუტერულ პროგრამებთან ადამიანს შანსი აღარ ჰქონდა.

საჭადრაკო კომპიუტერული პროგრამები დღევანდელ რეალობაში და მათი განვითარების პერსპექტივები.

ბოლო წლებში საგრძნობლად გაიზარდა საჭადრაკო პროგრამების რაოდენობა, მუდმივად იმართება საერთაშორისო ტურნირები საჭადრაკო პროგრამებს შორის, რაც უდიდესს ინტერესს იწვევს ჭადრაკის მოყვარულებში. საჭადრაკო პროგრამებმა დიდი როლი ითამაშეს პროფესიონალი და მოყვარული მოჭადრაკეების მომზადების პროცესში, ბოლო 20 წელიწადში საგრძნობლად შეიცვალა ჭადრაკის თამაშის სტილი უმაღლესი დონის მოჭადრაკეებს შორის, რაც განპირობებულია საჭადრაკო პროგრამების თამაშის სტილით.

ჭადრაკის არაპროგნოზირებადობის და პარტიის განვითარების შესაძლო სვლების ძალიან დიდი რაოდენობა არ იძლევა სრულყოფილი მოთამაშის შექმნის საშუალებას. კომპიუტერულ პროგრამებში ჩადებულია დებიუტის და ენდშპილის საუკეთესო ვარიანტები. მაგრამ ის, თუ როგორ თამაშობს პროგრამა პარტიის შუაში, განპირობებულია შეფასების ფუნქციით, რომელიც მუდმივად იცვლება, თუმცა მისი სრულყოფილი ფორმულა არ არსებობს. როდესაც დებიუტების ბაზა და ენდშპილების ბაზა იმდენად დიდი გახდება, რომ ისინი ერთმანეთს შეერწყმებიან, კომპიუტერი „დაიწყებს და მოიგებს“.

ლიტერატურა

Donald Knuth, Ronald W. Moore (1975). An analysis of alpha-beta pruning. Artificial Intelligence, Vol. 6, No. 4, pp 293–326. Reprinted in Donald Knuth (2000).

Shannon, Claude E. (1950), Programming a Computer for Playing Chess

Rybka Cluster - for the elite chess player. Rybkachess.com (February 1, 2011).

Hsu, Feng-hsiung (2002). “Behind Deep Blue: Building the Computer that Defeated the World Chess Champion”. Princeton University Press