Week 12 HIA

Terry Yin

November 27, 2014

1 State Graphs and Search trees

Chess-like Game can be a good example for reasoning. In any state of a chess game that is not the end of the game, there could be many possible next moves. Each of these possible next moves will lead the chess play into a next state. When linking all these states together, we will get a graph.

Sometimes, the graph looks like a tree. For example, in the game Reversi Wikipedia (2014*e*), the two players keep adding disks to the 8*8 board. The color of the already put disks might change, but there's only adding, not removing. So the game will always end after at most 60 moves. Although the same state of the game could be achieved by different path of state changing, because the game will end in limited moves any way, we can just ignore the duplicated states. Then the state graph looks just like a tree.

Sometimes, the graph is not a tree and have cycles. For example, in the game of chess, you can move the chesses back and forth again and again (unless there's extra rules). So the states can form a circle. And searching in the graph could end up in an infinite loop if there's circles in the graph.

Search tree is the way of searching or traversing the state graphs. When the problem is a tree (like Reversi), we don't have the keep the record of the nodes that have been searched already. If the problem is not a tree but a graph (like chess), we need to keep all the record that have been searched already to avoid circling search.

The basic methods of traversing a search tree are breadth first and depth first. The most well known breadth first search algorithm is Dijkstra's algorithm Wikipedia (2014*a*). When the search tree is very large, we need some trimming to reduce the nodes that we need to search. This will need us to implement some heuristics, which represents the evaluation of a nodes which is not the end nodes (Brookshear 2011, p.478). In a chess-like game, the often used algorithm for an optimized searching is the minmax algorithm Wikipedia (2014*c*), or it's simplified version, the negamax algorithm Wikipedia (2014*d*).

2 Cloud Computing and Green Computing

Cloud computing is the technology the virtualize computing resources utilizing the ability of Internet. Because of the virtualization, the uses in general doesn't need to care about where exactly are the physical machines they are using and how the services and data are distributed. It's also easier to scale up or down. Because cloud computing uses the computing resources for a larger pool, the usage rate of the computing resource is much higher than private servers. So it can reduce the amount of computers and may potentially consume less power.

Green computing is the study and pratice of being environment friendly in computing Wikipedia (2014*b*). As the need for computing increases the use of nature resources has been increasing as well. Building computers needs materials from the nature. The computing will consume power. The cooling of the machines will also consume power and other nature resources. One of the direction of going green is to use resources more efficiently.

Cloud computing services are often hosted in large data centers, which is like a big farm of computers. There are many way a data center could reduce in their use of nature resource. And they can do a lot better

than today.

> "I feel it's embarrassing, personally, that on average it will take more time and energy to get a collection of software from one data center to the next than it is to ship physical goods from on side of the planet to another. Hykes (n.d.)"

3 Intelligence and Simulated Intelligence

Intelligence is something possesed by a human, for example, the ability of logic, self-awareness and problem-solving. We can simulate some of the human intelligence using computer algorithms.

But a human mind can cheerfully handle many reasoning, different view points and romantic in the same time. It's very hard to be simulated. Typically, the simulated intelligence is good at only limitted things. For example, a chess playing algorithm is only good at chess playing. It doesn't know how to read the struggling expression on its human opponent's face. In artificial intelligence this is called a intelligent agent (Brookshear 2011, 462).

Traditional, a Turing test (Brookshear 2011, 465) is used to measure the quality of an implementation of artificial intelligence. But as turing test is not very practical, it's more of a historical story nice to learn in computer science than a real test. The AI studies nowadays focus more on specialized intelligent agents rather than simulating the human intelligence.

4 Trade-offs in Develop Mobile Applications

There are still some limitation with mobile devices compare to a desktop computer. Just to name a few:

- Battery life (it in general need to last at least one day by normal uses)
- Screen size is smaller
- Platform variation. All the major platforms, Android, iOS and Windows are all quite different.
- The computing resource is restricted by limited memory and cpu capacity.

Battery life If you wish the users to use your application often on their mobile device, you need to limit the computing resource your application uses. Otherwise when people found their mobile device becomes very hot or the battery dies too fast when running your application, they will stop using it. For example, until now, using the AngularJS Darwin & Kozlowski (2013) to develop web based application for mobile is probably a bad idea. Because frameworks like AngularJS use too much resouce.

Screen size It's relatively easier to develop for iOS, because there's not many screen size variations. There are too many different mobile devices base on Android system and too many different screen sizes. In general, HTML is enough to create graphical user interface for mobile device. But if you want find tuned UI, you will need to have specific design for each screen size.

Platform variation Typically, people use objective-C or Swift to develop iOS application, use Java for Android and use C# for Windows. It make sharing solutions between these platforms very hard. But there's still something we can do to make the situation better. The different platforms could at least share the same set of tests. Testing tools like Calabash (http://calaba.sh) makes it possible to do cross platform testing. So you run the automated test once and run it on every devices. There's also open source project like the "mobile chrome apps"(https://github.com/MobileChromeApps/mobile-chrome-apps) project trying to make it possible to create the apps once and run it on both iOS and Android.

Memory and CPU limitation The limitation is becoming already a lot better. Some phones already have 4G memory. But still, a mobile device doesn't have advantage in computing reources. So when possible, we want to keep the computing job on the server side. And because some mobile apps are continuously on

the device just like an embedded software, the developer also need to pay extra attention to the accumulative resource consumption. For example, a little memory leak in the app might create big trouble if it has been runing for days on the mobile device.

References

Brookshear, J. G. (2011), *Computer science: an overview*, Paul Muljadi.

Darwin, P. B. & Kozlowski, P. (2013), *AngularJS web application development*, Packt Publishing.

Hykes, S. (n.d.), 'What is docker?', https://www.youtube.com/watch?v=ZzQfxoMFH0U#t=245. Accessed: 2014-11-17.

Wikipedia (2014*a*), 'Dijkstra's algorithm — wikipedia, the free encyclopedia'. [Online; accessed 26-November-2014].

Wikipedia (2014*b*), 'Green computing — wikipedia, the free encyclopedia'. [Online; accessed 26-November-2014].
URL: *http://en.wikipedia.org/w/index.php?title=Green_computing&oldid=635506405*

Wikipedia (2014*c*), 'Minimax — wikipedia, the free encyclopedia'. [Online; accessed 26-November-2014].
URL: *http://en.wikipedia.org/w/index.php?title=Minimax&oldid=634666658*

Wikipedia (2014*d*), 'Negamax — wikipedia, the free encyclopedia'. [Online; accessed 26-November-2014].
URL: *http://en.wikipedia.org/w/index.php?title=Negamax&oldid=632111318*

Wikipedia (2014*e*), 'Reversi — wikipedia, the free encyclopedia'. [Online; accessed 26-November-2014].
URL: *http://en.wikipedia.org/w/index.php?title=Reversi&oldid=634714760*