Institut for Matematik og Datalogi
Syddansk Universitet

December 5, 2008
JFB

# Introduction to Computer Science
# E08 – Week 13

## Instruktorater

**Hjælp dine medstuderende, dygtiggør dig selv og få penge for det. Søg I dag!**

Der er ofte gode muligheder for at få et instruktorat, selvom man ikke er langt henne i studiet. Se nærmere information på

`http://www.jobs.sdu.dk/vis_stilling.php?id=4803&lang=da`

Hvis du allerede er ansat som instruktor i efteråret 2008, leverer du ansøgning om tildeling af timer ind på IMADAs sekretariat (se opslaget vedr. bilag mm.)

Hvis du har spørgsmål, så henvend dig gerne p IMADA.

ANSGNINGSFRIST: **15. december 2008 kl 12:00**.

## Lecture, December 1

We covered the first three sections of chapter 11 and up through the subsection on "Representing and Manipulating Knowledge" in the fourth section.

## Lecture, December 8

We will continue with chapter 11 in the textbook.

## Lecture, December 15

We will discuss the course in relation to the rest of your studies.

## Discussion section: first in week 51, in the Terminal Room

Continue on your lab from week 48 on hash funcitons, in the same groups, if possible. The following is a repeat of what appeared on that lab description:

1. Design three hash functions and program them in Java (or Maple).
2. Test your hash functions. Create an array of length $m+1$ which will count how many random large integers are hashed to each of the distinct values between zero and $m$. Look at the results for some relatively small values of $m$ (for example, $m = 4, 8, 12, 20, 50, 100$)
3. With at least four different values of $m$ and some different length large strings (integers) which are hashed, check how many values you need to hash before you get a collision. Does it behave as one would predict with the Birthday Paradox?
4. With at least four different values of $m$ and some different length large strings, check how even your distribution is? What is the difference between the largest number of strings hashing to any particular value and the smallest number? How does this seem to depend on the number of strings you hash?
5. With your hash functions, how would you find collisions (two strings that hash to the same value) when $m$ is very large (say $2^{80}$)? Are your hash functions cryptographically secure?

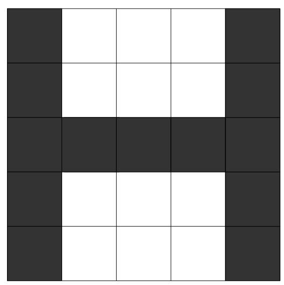
## Discussion section: second in week 51

In groups of three or four, discuss the following problems (some are from the textbook):

In the first two problems we are going to construct a network that will recognize specific (simple) patterns. Our basic model for an image is a $5 \times 5$ grid whose different positions are numbered from 1 to 25. Thus, we can think about these positions as 25 input values to our network, denoted by $x_1, \ldots, x_{25}$. These inputs can take the values 0 or 1, only. A 0 value corresponds to a white grid point, a 1 value codes a black grid point.

| | | | | |
|---|---|---|---|---|
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
| $x_6$ | ... | ... | ... | ... |
| ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... |
| ... | ... | ... | ... | $x_{25}$ |

- Construct a perceptron taking the 25 input values $x_1, \ldots, x_{25} \in \{0, 1\}$ that precisely recognizes the following pattern of the latter **H**:

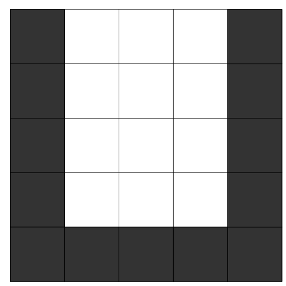
  

  Here, to construct means that you should find suitable real values $w_1, \ldots, w_{25}$ and $b$ such that the resulting perceptron network gives output 1 if and only if the inputted values $x_1, \ldots, x_{25}$ precisely represent the above pattern.

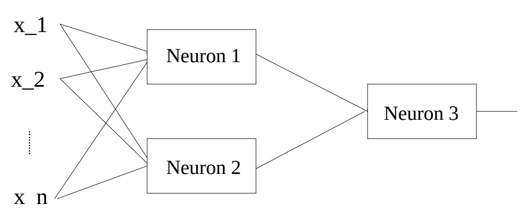
- Now we consider a second pattern. We would like to build a ne twork that is able to recognize the letter **H** from before and the letter **U**:

  a) Why is it impossible to solve that problem by just using a single perceptron network? In your answer, suppose that a perceptron network exists that solves the problem. Then argue about how that network has to arrange its weights in order to accept the two above patterns but not any other pattern.

  b) Solve the above pattern recognition problem by using a more advanced network structure. The network should still consist of perceptron neurons. However, now there is a first layer with two



perceptrons in it each of which is connected to the 25 inputs and having its own set of weights and its own threshold. Then, the results of these two perceptron neurons are connected in a second layer by a third perceptron neuron, again taking new weight values that linearly combine the results of the first two neurons and computing its result using a third threshold value.




All edges in the above network get their own (may be similar) weight values.

- Questions 1, 2, 3, 4, and 7 on page 533.
- Questions 4, 5, and 6 on page 546.
- Questions 1, 2 and 3 on page 560.

## Remember the assignment due 14:15, December 18

You make pick up the graded assignment in my office any time after 9 AM, January 2 or on January 5, 6, or 7. If you have to do a retry, it will be due 14:15 on January 19. Note that I will be out of town from January 8 through January 18, but Arun says that you can send him an e-mail and arrange a time to talk with him that way.