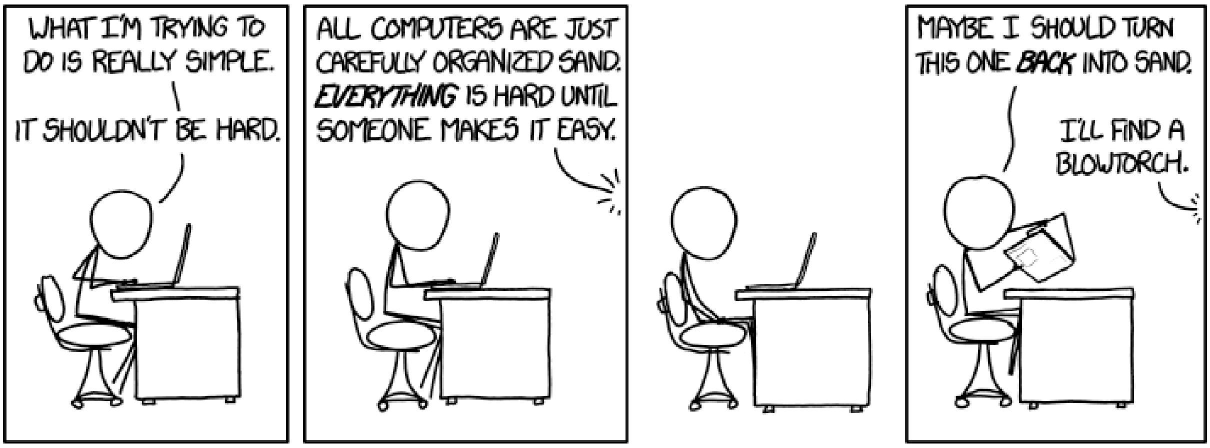
CITS3001 Mid-semester Test 2014

Fifty minutes

Answer all four questions

Total marks 60



When showing the operation of an algorithm, include enough detail to make it clear that you understand how *the algorithm* solves the problem.

Question 1: the Boyer-Moore algorithm for pattern-matching (15 marks)

Briefly describe the principles, operation, and performance issues of the Boyer-Moore algorithm for pattern-matching on strings.

Illustrate your answer using the DNA text and pattern in Fig. 1. Note that for DNA sequences, the alphabet is ACGT, i.e. only four characters.

T = TAGTCGTGCGCGAAATTCGTGCGCGA

P = TCGTGCGCG

Fig. 1: DNA text and pattern for Question 1.

Make sure that you show

- the complete pre-calculated tables for all heuristics;
- the shifts that are considered by the algorithm, the shifts that are skipped, and why; and
- all shifts that are returned as successful matches.

Question 2: the Travelling Salesman Problem **(15 marks)**

Briefly describe the principles, operation, and performance issues of any *one* commonly-used approximation algorithm for the Travelling Salesman Problem.

Illustrate your answer using the set of cities in Fig. 2.

	A	B	C	D	E
A	−	40	10	20	12
B	40	−	35	50	32
C	10	35	−	30	16
D	20	50	30	−	19
E	12	32	16	19	−

Fig. 2: Cities and the distances between them for Question 2.

Make sure that you show all relevant operational details of your chosen algorithm.

Question 3: Iterative-deepening depth-first search **(15 marks)**

Briefly describe the principles, operation, and performance issues of iterative deepening depth-first search.

Illustrate your answer using the road map of Nerdvana in Fig. 3. Find the route from Geek to Nerd that goes through the fewest intermediate cities.

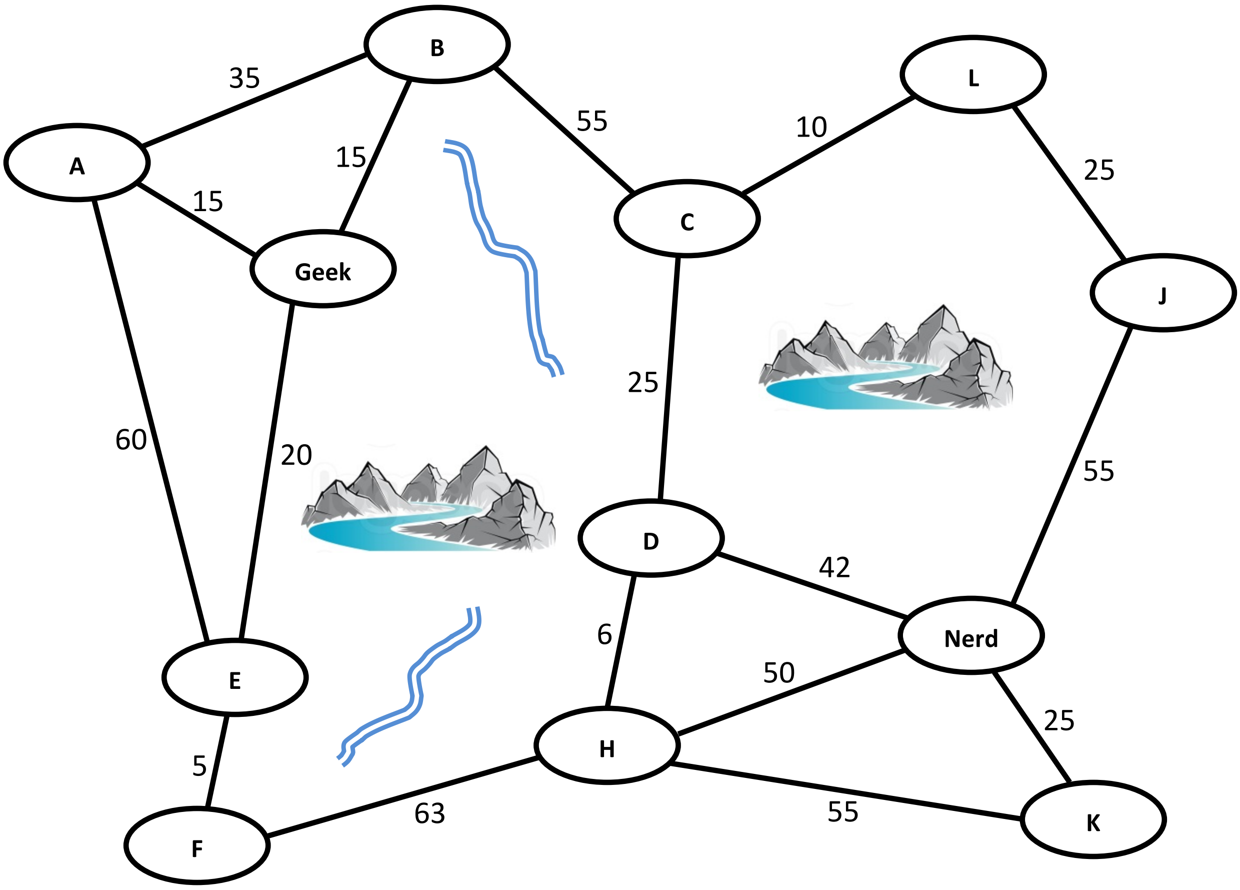


Fig. 3: Road map of Nerdvana for Questions 3 and 4.

Make sure that you show which nodes are expanded, in what order, and why.

Note that within one iteration, when you have expanded a node X at depth k, there is no need to expand X again at any depth $j \geq k$.

Question 4: *A** **(15 marks)**

Briefly describe the principles, operation, and performance issues of *A**.

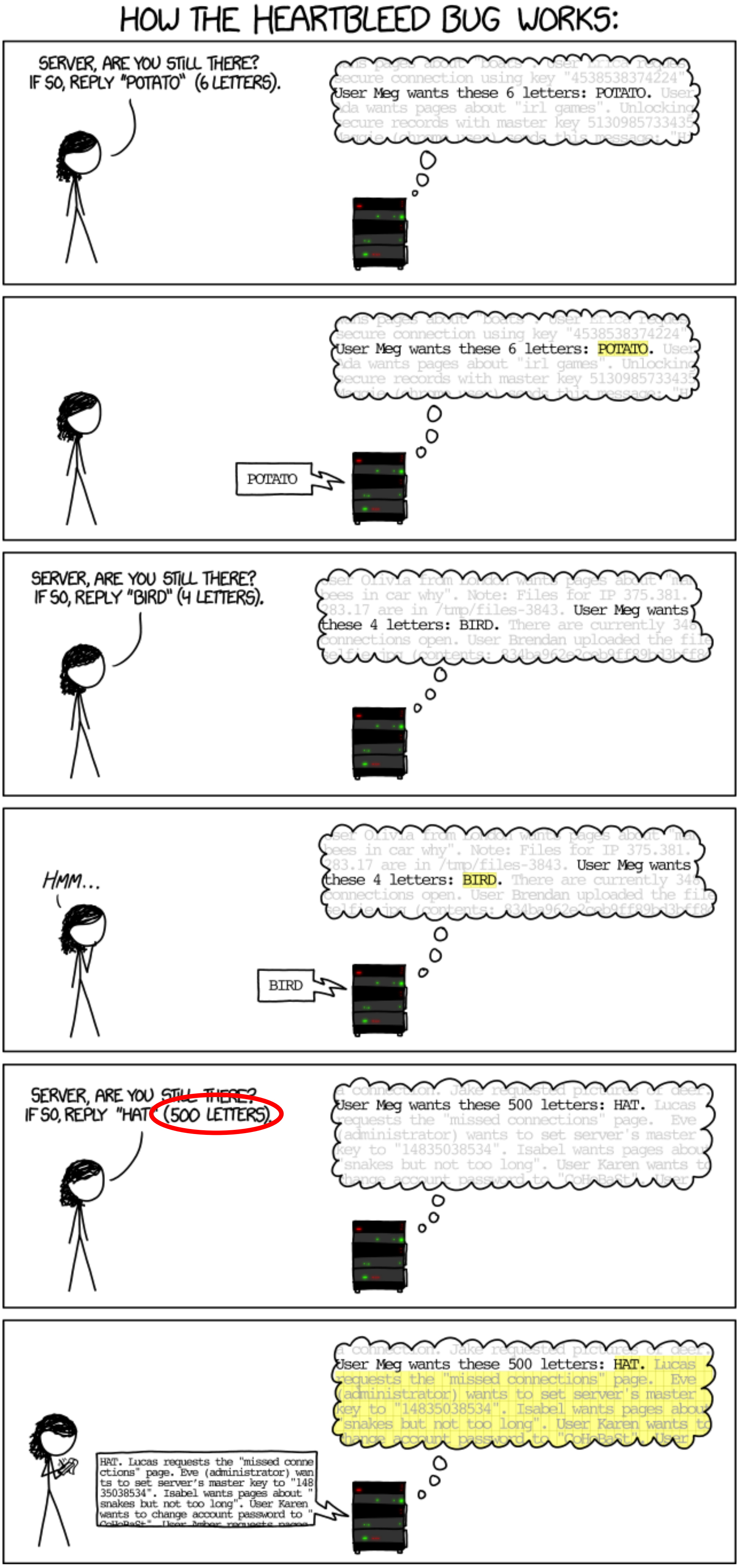
Illustrate your answer using the road map of Nerdvana in Fig. 3 and the table of straight-line distances in Fig. 4. Find the route from Geek to Nerd that requires the least total distance.

Geek	**95**	**F**	**100**
A	**125**	**H**	**45**
B	**100**	**J**	**50**
C	**60**	**K**	**25**
D	**40**	**L**	**70**
E	**90**	**Nerd**	**0**

Fig. 4: Straight-line distance from each city to Nerd for Question 4.

Make sure that you show which nodes are expanded, in what order, and why.

Note that when you have expanded a node X at $g(X) = k$, there is no need to expand X again at any $g(X) \geq k$.


HOW THE HEARTBLEED BUG WORKS:
SERVER, ARE YOU STILL THERE? IF SO, REPLY "POTATO" (6 LETTERS).
User Meg wants these 6 letters: POTATO.
POTATO
SERVER, ARE YOU STILL THERE? IF SO, REPLY "BIRD" (4 LETTERS).
User Meg wants these 4 letters: BIRD.
HMM...
BIRD
SERVER, ARE YOU STILL THERE? IF SO, REPLY "HAT" (500 LETTERS).
User Meg wants these 500 letters: HAT.
HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to "

Solutions

Question 1

Need to mention:

- same basic system as the naïve algorithm;
- matches pattern right-to-left;
- uses the bad character heuristic;
- and the good-suffix heuristic;
- beats linear in good cases (or avoids some characters completely).

T = TAGTCGTGCGCGAAATTCGTGCGCGA

P = TCGTGCGCG

x[y] below for *γ(j)* means *x* is the basic shift (P. 19), *y* is the advanced (P. 21).

b	*λ(b)*	*j*	*γ(j)*
A	**0**	**8**	**2[4]**
C	**8**	**7**	**2[6]**
G	**9**	**6**	**2**
T	**4**	**≤ 5**	**9**

Matching occurs from right-to-left at each shift.

T[9] = C, so the suggested shifts are 9–8 = 1 and 1

T[8–10] match, so the suggested shifts are 6–4 = 2 and 2

T[4–12] match, so the suggested shifts are 9 and 9

T[21] matches, so the suggested shifts are 8–4 = 4 and 2[4]

T[17–25] match, so the suggested shifts are 9 and 9 and we are done

Question 2

Various

Question 3

Need to mention five of:

- sequence of searches;
- each one is depth-limited;
- the limit increases in each iteration;
- space is linear in depth;
- optimal and complete;
- some nodes are inspected multiple times.

X* indicates that Node X has already been expanded.

Limit = 0, Geek is not a goal

Limit = 1, G(ABE)

Limit = 2, G(

A(BGE)

B(AGC)

E(AGF))

Limit = 3, G(

A(B(AGC)G*E(AGF))

B(A*G*C(BDL))

E(A*G*F(EH)))

Limit = 4, G(

A(B(A*G*C(BDL))G*E(A*G*F(EH)))

B(A*G*C(B*D(Nerd)L))

E)

So the solution with the fewest cities is GBCDN. (GEFHN is also acceptable.)

Question 4

Need to mention:

- maintains a pool/queue of unexpanded nodes, from the start node;
- lowest-cost node is selected next;
- cost of *n* is actual cost of start-to-*n* plus estimated cost of *n*-to-goal;
- provably optimal with admissible heuristic;
- good space behaviour requires *SMA**.

c	*g(c)*	*h(c)*	*f(c)*
Geek	0	95	95

Expand G

c	*g(c)*	*h(c)*	*f(c)*
E	20	90	110
B	15	100	115
A	15	125	140

Expand E, ignore G and A, sort by *f*

c	*g(c)*	*h(c)*	*f(c)*
B	15	100	115
F	25	100	125
A	15	125	140

Expand B, ignore G and A, sort by *f*

c	*g(c)*	*h(c)*	*f(c)*
F	25	100	125
C	70	60	130
A	15	125	140

Expand F, ignore E, sort by *f*

c	*g(c)*	*h(c)*	*f(c)*
C	70	60	130
H	88	45	133
A	15	125	140

Expand C, ignore B, sort by *f*

c	*g(c)*	*h(c)*	*f(c)*
H	88	45	133
D	95	40	135
A	15	125	140
L	80	70	150

Expand H, ignore F, replace D, sort by *f*

c	*g(c)*	*h(c)*	*f(c)*
D	94	40	134
N	138	0	138
A	15	125	140
L	80	70	150
K	143	25	168

Expand D, ignore C and H, replace N, sort by *f*

c	*g(c)*	*h(c)*	*f(c)*
N	136	0	136
A	15	125	140
L	80	70	150
K	143	25	168

So the solution with the least distance is GEFHDN.