cuACS Project

Algorithm Design Document

Team 070 – Segmentation Fault

Weihang Chen

Liyongshi Chen

Richard Xia

Xiran Zhou

Submitted to:

Dr. Christine Laurendeau

COMP3004 Object-Oriented Software Engineering

School of Computer Science

Carleton University

Table of Content:

1. Introduction

1.1. Overview of the Algorithm

The design goal of the animal-client matching (ACM) algorithm is to generate an optimal set of animal-client matches, for which clients' requirements and preferences should be satisfied as much as possible with more sheltered animals being adopted. To achieve this goal, we constructed our ACM algorithm into two sub algorithms:

1. Profile matching (begin algorithm), where points are calculated using a set of rules to determine the compatibility of each animal-client pair.

2. Adoption maximization (end algorithm), where a priority queue is established according to the number of possible matches for each client, as well as for each animal available for adoption.

The ACM algorithm should output a set of animal-client pairs, such that the best interests of all sheltered animals are considered in addition to clients' desires.

1.2. Overview of Document

The purpose of this document is to describe our ACM algorithm design in such a way that it is understood by all interested parties. This document is divided into multiple sections including an introduction, the non-physical attributes used by the algorithm and their valid range, the rules for matching, the way to compute the optimal set of matches, and a summary of the document, respectively.

2. Non-physical Attributes

Non-physical attributes of an animal, i.e. temperament and habits, are traits that cannot be measured directly (Unlike the weight of an animal). The non-physical attributes used in the ACM algorithm as well as their valid range of values are listed in Table 1 below.

Table 1 – Non-physical Attributes Used to Compute Animal and Client Matches

Non-physical attributes	Valid Range of Values
1. Aggressivity ● Testy ● Middle ● Mild	{-1, 0, 1}, with -1 being the least desirable match and 1 being the most desirable match.
2. Personality ● Active ● Quiet ● Shy ● Lazy ● Friendly	{1/5, 2/5, 3/5, 4/5, 1}, with 1/5 being the least desirable match and 1 being the most desirable match.
3. Sociality ● Gregarious ● Solitary ● Gregarious/solitary	{1, 2, 3}, with 1 being the least desirable match and 3 being the most desirable match.
4. Food Preferences ● Low sugar ● Low salt ● Uncooked ● Low fat ● Only Liquid	$\{x \in R \mid 0 \leq x \leq 1\}$, with 0 being the least desirable match and 1 being the most desirable match.
5. Service Ability ● Guide dog ● Animal therapy ● Children care ● None	{0, 1}, with 0 being the least desirable match and 1 being the most desirable match.
6. Interaction ● Simple gesture commands ● Simple verbal commands	{0, 1/4, 2/4, 3/4, 1}, with 0 being the least desirable match and 1 being the most desirable match.

● Complex verbal commands ● None	
7. Psychological Disorder ● Accident ● Abandoned ● Maltreated ● None	{1/4, 2/4, 3/4, 1}, with 1/4 being the least desirable match and 1 being the most desirable match.
8. Activity Habit ● Active at daytime ● Active at night ● Never active ● Always active	{0, 2/5, 3/5, 1}, with 0 being the least desirable match and 1 being the most desirable match.
9. Days Since Sheltered ● Integer of days	{x ∈ R | 0 ≤ x ≤ 1}, with 0 being the least desirable match and 1 being the most desirable match.
10. Adaptability ● Good ● Need some time ● Need a long time	{1, 2, 3}, with 1 being the least desirable match and 3 being the most desirable match.
11. Resource ● Previous owners ● Homeless ● Laboratory ● Rescued	{-1, -3/4, 1/4, 1/2, 3/4, 1}, with -1 being the least desirable match and 1 being the most desirable match.
12. Necessities Needs ● 0-200 ● 200-400 ● 400-600 ● More that 600	{x ∈ R | 0 ≤ x ≤ 1}, with 0 being the least desirable match and 1 being the most desirable match.

3. ACM Matching Rules

3.1. Overview of the ACM Matching Rules

This section will focus on the begin algorithm of the ACM – Profile Matching sub algorithm. It implements a unique set of rules to compute animal-client matches. The rules are defined according to the attributes contained in both animal and client profiles. Weights are assigned to each animal-client attribute pair and formulas are used to compute the resulting points. The output of this sub algorithm is a set of animal-client pairs, which will be used as an input parameter of the end algorithm – Adoption Maximization sub algorithm.

The essence of ACM matching rules can be expressed as the following formula:

$$\{S \mid S = \sum_{i=1}^{12} w_i * f(i), S > 6\},$$

where S is the total matching points of an animal-client pair, w_i is the weight assigned to each attribute, and f(i) is the formula used to calculate point subtotals for each attribute. The pair (Animal_i, Client_i) would be considered as a valid desirable match only when the total matching points S_i(Animal_i, Client_i) is greater than 6. All desirable matches are then selected and put into a set. This set is defined as the output of the Profile Matching sub algorithm.

3.2. Profile Matching Sub Algorithm

This sub section defines and explains the set of rules used to determine animal-client matches based on their profile and preferences. Tables and pseudo codes will be provided to list, explain, and justify the rules used to compute matches; Graphs will be provided to demonstrate the flow of the algorithm and charts will be provided to illustrate the details of certain rules where appropriate.

Table 2 – Animal Attributes and Associated Client Information Dependencies

Non-physical Attribute	Associated Client Information	Description
1. Aggressivity	Has children; Age	Clients with young children should be matched with less aggressive animals; Clients who are over 60 should be matched with less aggressive animals.
2. Personality	Preferences	Clients can choose more than one personality traits according to their preferences.
3. Sociality	Hours willing to spend on pets per day	Clients who aren't able to spend much time on animals should be more likely to get solitary animals, and vice versa.
4. Food Preferences	Monthly income	Clients with lower income should be matched with animals that are less picky to feed. The richer a client is, the more expensive/demanding food he or she is able to afford.
5. Service Ability	Special requirements	Clients who require service animals should be considered as the top priority among all matching criteria.
6. Interaction	Has children; Age	Clients who have children or who are older are more likely to be matched with animals that have high interaction capabilities.
7. Psychological Disorder	Animal care experience	Clients who have previous experience taking care of animals should be matched with animals that require special care, i.e., animals with psychological disorder.
8. Activity Habit	Daily routine	Clients who work during the day and sleep at night are likely to be matched with diurnal animals. Clients who works overnight and sleep during the day are likely to

		be matched with nocturnal animals. Active animals always have lower rank than never active ones regardless of client's daily routine.
9. Days Since Sheltered	N/A	First in first out principle. Animals that have been sheltered for a relatively long period should be adopted first.
10. Adaptability	Hours willing to spend on pets per day	Clients who aren't able to spend much time on animals should be more likely to get more adaptive animals, and vice versa.
11. Resource	Animal care experience	Clients who have previous experience taking care of animals should be matched with vulnerable animals, i.e., animals that are rescued from abuse or from laboratories.
12. Necessities Needs	Monthly income	Clients who have higher income should be matched with animals that require more expensive necessities (i.e., cages, aquariums, ponds, etc).

Table 3 – Rules Used for Matching Animal and Client Profile

Formula: $S = \sum_{i=1}^{12} w_i * f(i)$,

where S is the total matching points of an animal-client pair, w_i is the weight assigned to each attribute, and f(i) is the formula used to calculate point subtotals for each attribute, as stated in the previous section.

Note: "C" refers to a Client object;

"A" refers to an Animal object;

"S_i" refers to the matching points for index *i*.

The operator "." is used to access an attribute of the target object.

1. Aggressivity: f(1) = A.aggrPoints; w_1 = 1; S_1 = 1* A.aggrPoints		
Attr. value	*Rule*	*Reason for the rule*
Testy	If C.hasChildren==true or C.age>60 Then A.aggrPoints = -1; Else A.aggrPoints = 1	Vulnerable group, including young children and elderly people, should be considered first. This group of clients are likely to be matched with animals that have gentle temper.
Middle	A.aggressivity = 0;	
Mild	If C.hasChildren==true or C.age>60 Then A.aggrPoints = 1; Else A.aggrPoints = -1	

2. Personality: f(2) = (1/5)*A.persPoints; w_2 = 1; S_2 = 1* (1/5)*A.persPoints		
Attr. value	*Rule*	*Reason for the rule*
Active	Each value worth 1 point; Client can choose more than one value; A.persPoints = the total number of choices that a client has selected. Note: f(2)=(1/5)* A.persPoints.	This rule is to consider the preferences of a client and try to give a match that is relatively favorable to the client.
Quiet		
Shy		
Lazy		
Friendly		

3. Sociality: f(3) = (1/3)*A.socPoints; $w_3 = 1$; $S_3 = 1*(1/3)*$ A.socPoints		
t= number of hours available for pets (hours/day); Possible values for t: 2, 4, 6 (hours/day).		
Attr. value	*Rule*	*Reason for the rule*
Gregarious	If t==2 then A.socPoints =1; If t==4 then A.socPoints =2; If t==6 then A.socPoints =3;	This rule tries to match solitary animals to clients that have busy schedules and gregarious animals to less busy clients.
Solitary	If t==2 then A.socPoints =3; If t==4 then A.socPoints =2; If t==6 then A.socPoints =1;	
Gregarious or Solitary	If t==2 then A.socPoints =2; If t==4 then A.socPoints =3; If t==6 then A.socPoints =2	

4. Food Preferences: f(4) = A.fpPoints; $w_4 = 1$; $S_4 = 1*$A.fpPoints		
Attr. value	*Rule*	*Reason for the rule*
Low Sugar Low Salt Uncooked Low fat Only Liquid	s =client's salary; n = number of selected choices. A.fpPoints = min {1, ((s – 2000)/500) / n}	The more selections a client makes in Food Preferences, the more cost it is to feed this animal. This rule is to make sure that a client is capable of feeding the animal.

5. Service Ability: f(5) = A.servPoints; w_5 = 1 or 50; $S_5 = w_5$ * A.servPoints		
Attr. value	*Rule*	*Reason for the rule*
Guide dog	If C.requirment == Guide Dog Then A.servPoints = 1, w_5 = 50; Else A.servPoints = 0, w_5 =1;	This rule is to ensure the special requirements of a client is satisfied with top priority by making the weight w very large. Animals with service abilities are saved for clients in need.
Animal therapy	If C.requirment == Animal Therapy Then A.servPoints = 1, w_5 = 50; Else A.servPoints = 0, w_5 =1;	
Children care	If C.requirment == Children Care Then A.servPoints = 1, w_5 = 50; Else A.servPoints = 0, w_5 =1;	
None	If C.requirment == None Then A.servPoints = 1, w_5 =1; Else A.servPoints = 0, w_5 =1;	

6. Interaction: f(6) = (1/3)*A.interactPoints; w_6 = 1; S_6 = 1* (1/3)*A.interactPoints		
Attr. value	*Rule*	*Reason for the rule*
Simple gesture commands	If C.hasChildren==true or C.age>60 Then A.interactPoints = 1; Else A.interactPoints = 0;	This rule tends to match vulnerable people with animals that are more intelligent.
Simple verbal commands	If C.hasChildren==true or C.age>60 Then A.interactPoints = 2; Else A.interactPoints = 0;	

Complex verbal commands	If C.hasChildren==true or C.age>60 Then A.interactPoints = 3; Else A.interactPoints = 0;	
None	If C.hasChildren==true or C.age>60 Then A.interactPoints = 0; Else A.interactPoints = 3;	

7. Psychological Disorder: f(7) = (1/4)*A.psydPoints; $w_7 = 1$; $S_7 = 1*(1/4)*A.psydPoints$		
Attr. value	*Rule*	*Reason for the rule*
Accident	If C.animal_care_experience==true Then A.psydPoints = 4; Else A.psydPoints = 0;	This rule is to match animals with potential psychological disorders (exposed to trauma or previously abused) to clients that have experience in animal care. In this way, animals that need special care will be in good hands.
Abandoned	If C.animal_care_experience==true Then A.psydPoints = 3; Else A.psydPoints = 1;	
Maltreated	If C.animal_care_experience==true Then A.psydPoints = 2; Else A.psydPoints = 2;	
None	If C.animal_care_experience==true Then A.psydPoints = 1; Else A.psydPoints = 4;	

8. Activity Habit: f(8) = A.actPoints; $w_8 = 1$; $S_8 = 1$* A.actPoints		
Attr. value	*Rule*	*Reason for the rule*
Active at daytime	If C.nightPerson == true Then A.actPoints = 0; Else A. actPoints = 1;	For a client who works in the day and sleeps at night, diurnal animals are more desirable. For a client who works overnight and sleeps during the day, nocturnal animals are more desirable. In this way, the client is less likely to be disturbed by the animal because of different circadian rhythms. An always active animal gets the least points because it disturbs a clients' schedule regardless.
Active at night	If C.nightPerson == true Then A.actPoints = 1; Else A. actPoints = 0;	
Never active	A.actPoints = 0.6;	
Always active	A.actPoints = 0.4;	

9. Day Since Sheltered: f(9) = A.dssPoints; $w_9 = 1$; $S_9 = 1$*A.dssPoint		
Attr. value	*Rule*	*Reason for the rule*
Integer of days	If A.day_since_sheltered<15 Then A.dssPoints = 1; Else A.dssPoints = min{1, ln((day-15) /5) +1}	First in first out. When an animal spends more than 15 days in the shelter, it should be considered first for adoption.

10. Adaptability: f(10) = (1/3)*A.adaptPoints; $w_2 = 1$; $S_2 = 1*$ (1/3)*A.adaptPoints		
t= number of hours available for pets (hours/day); Possible values for t: 2, 4, 6 (hours/day).		
Attr. value	*Rule*	*Reason for the rule*
Good	If t==2 then A.adaptPoints =3; If t==4 then A.adaptPoints =1; If t==6 then A.adaptPoints =1;	For clients who don't have much free time to take care of their pets, animals with good adaptabilities are likely to be a good match. For clients who are able to spend more time on their pets, less adaptive animals are more likely to be assigned.
Need some time	If t==2 then A.adaptPoints =1; If t==4 then A.adaptPoints =3; If t==6 then A.adaptPoints =2;	
Need a long time	If t==2 then A.adaptPoints =1; If t==4 then A.adaptPoints =2; If t==6 then A.adaptPoints =3;	

11. Resource: f(11) = (1/4)*A.resourcePoints; $w_{11} = 1$; $S_{11} = 1*$ (1/4)*A.resourcePoints		
Attr. value	*Rule*	*Reason for the rule*
Previous owner	If C.animal_care_experience==true Then A.resourcePoints = 1; Else A.resourcePoints = 4;	This rule is to match animals with trauma or unfortunate experience to clients that have experience in animal care. In this way, animals that need special care will be in good hands.
Homeless	If C.animal_care_experience==true Then A.resourcePoints = 2; Else A.resourcePoints = 3;	
Laboratory	If C.animal_care_experience==true Then A.resourcePoints = 3; Else A.resourcePoints = 2;	

Rescued	If C.animal_care_experience==true Then A.resourcePoints = 4; Else A.resourcePoints = 1;	

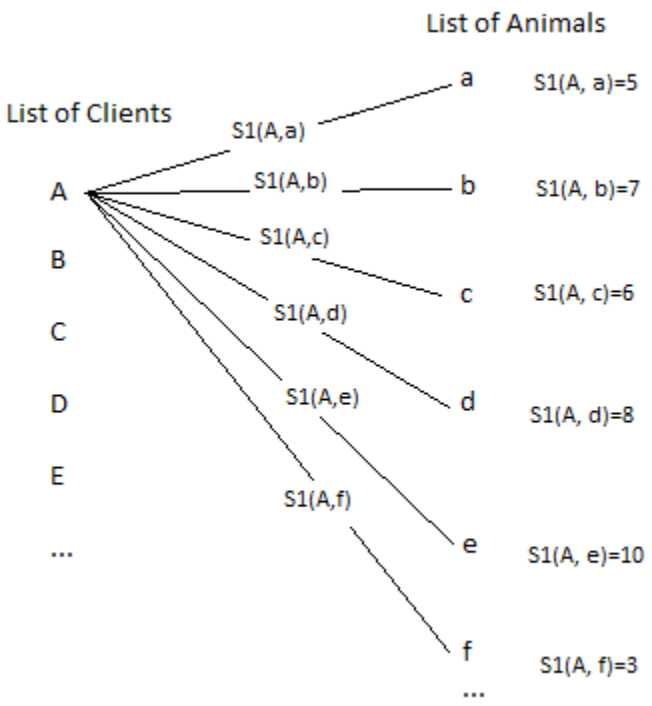
12. Necessities Needs: f(12) = A.nessPoints; w_{12} = 1; S_{12} = 1*A.nessPoints			
Attr. value	*Income_index*	*Rule*	*Reason for the rule*
0-200	1	Income_index: investment ability; N =min{1, ((salary-2000)%2000)} A.nessPoints = 1 – abs(income_index - N)*(1/4)	Some animals may require much more money on necessities than others. Clients who are able to afford the expense are matched with high-cost animals.
200 - 400	2		
400 -600	3		
> 600	4		

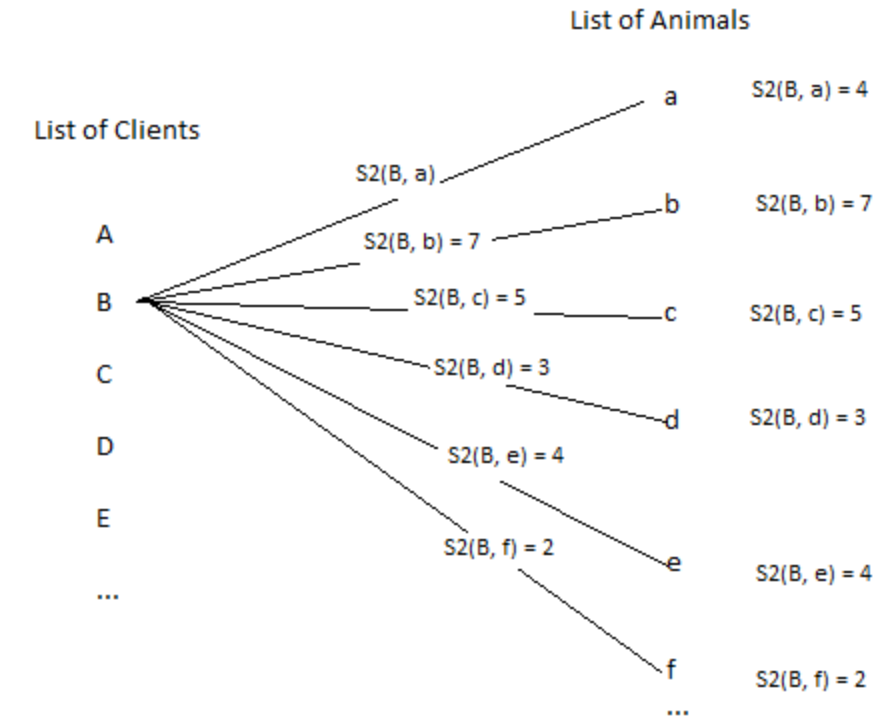
Graph 1 – Flow of the Profile Matching Sub Algorithm

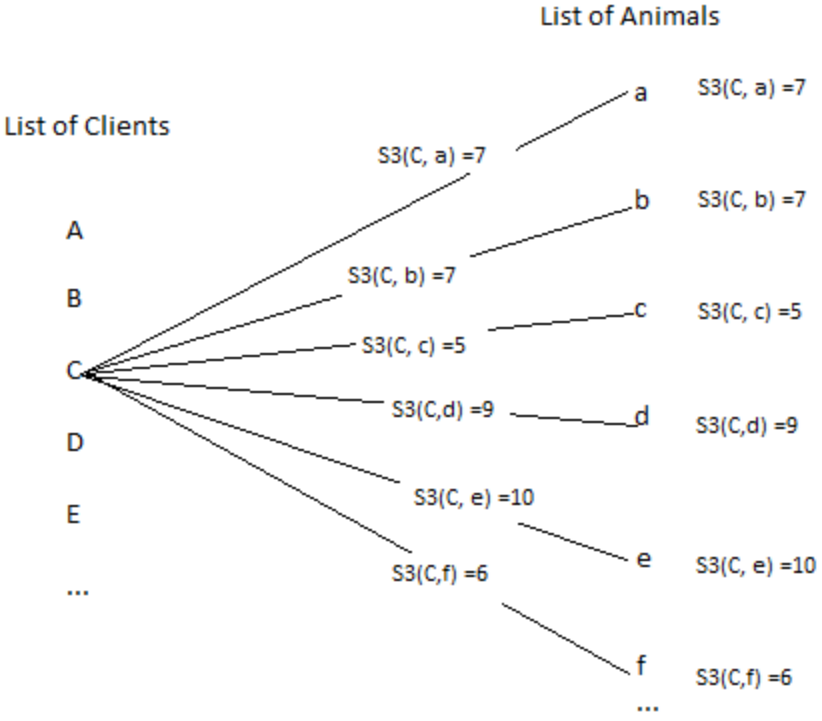
A sequence of graphs is constructed in this subsection to show how the Profile Matching algorithm is executed by giving a simple example. The execution steps are as follows: first, a list of Clients l_client = {A, B, C, D, E, ...} is extracted from the Human Information Database System and a list of Animals l_animal = {a, b, c, d, e, ...} is extracted from the Animal Information Database System. Second, each Client element l_client[i] from the Client List traverses the Animal List and compute S_i. Once all clients traversed the Animal List, select client-animal pairs (l_client[i], l_animal[i]) where S_i(l_client[i], l_animal[i])>6. This set of pairs is defined as the output of the Profile Matching algorithm.

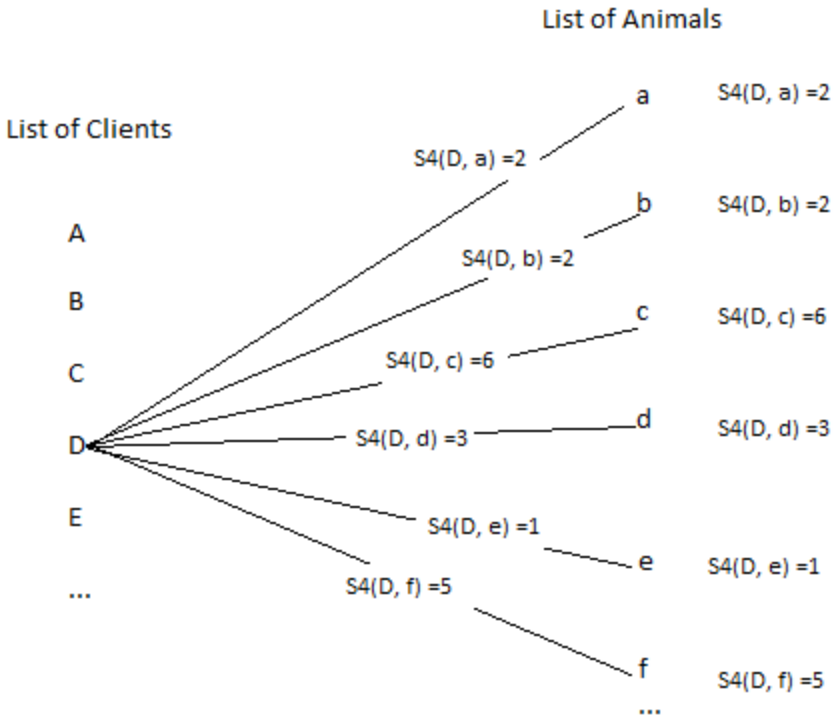
The sequence of graphs below illustrates the flow of Profile Matching algorithm execution.

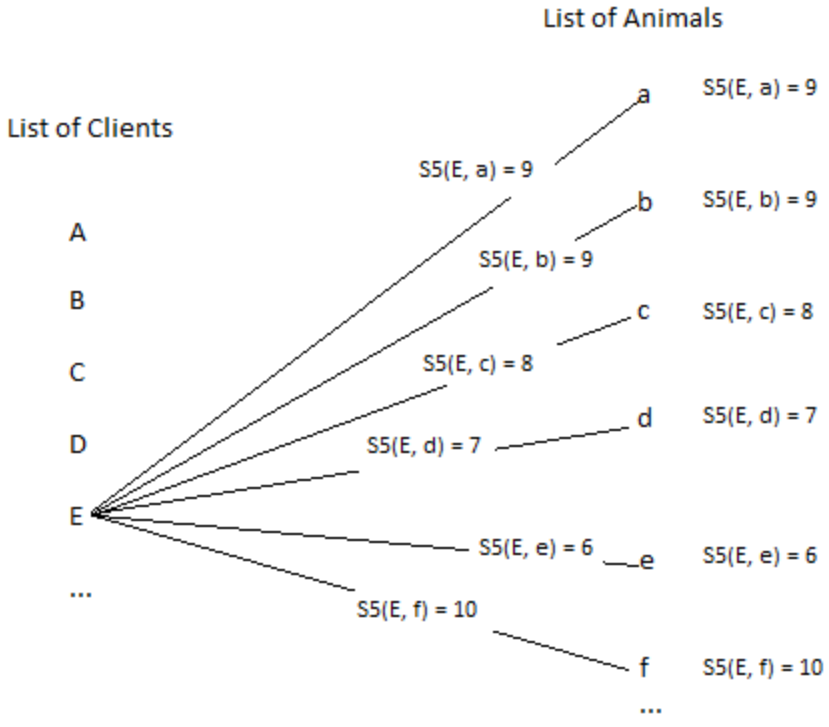
Stage 1: Each client *i* traverses the list of animals and compute point subtotal S_i.




List of Animals
List of Clients
A
B
C
D
E
...
S2(B, a)
S2(B, b) = 7
S2(B, c) = 5
S2(B, d) = 3
S2(B, e) = 4
S2(B, f) = 2
a
b
c
d
e
f
...
S2(B, a) = 4
S2(B, b) = 7
S2(B, c) = 5
S2(B, d) = 3
S2(B, e) = 4
S2(B, f) = 2


List of Animals
List of Clients
A
B
C
D
E
...
S3(C, a) =7
S3(C, b) =7
S3(C, c) =5
S3(C,d) =9
S3(C, e) =10
S3(C,f) =6
a
b
c
d
e
f
...
S3(C, a) =7
S3(C, b) =7
S3(C, c) =5
S3(C,d) =9
S3(C, e) =10
S3(C,f) =6


List of Animals
List of Clients
A
B
C
D
E
...
a
b
c
d
e
f
...
S4(D, a) =2
S4(D, b) =2
S4(D, c) =6
S4(D, d) =3
S4(D, e) =1
S4(D, f) =5
S4(D, a) =2
S4(D, b) =2
S4(D, c) =6
S4(D, d) =3
S4(D, e) =1
S4(D, f) =5


List of Animals
List of Clients
A
B
C
D
E
...
a
b
c
d
e
f
...
S5(E, a) = 9
S5(E, b) = 9
S5(E, c) = 8
S5(E, d) = 7
S5(E, e) = 6
S5(E, f) = 10
S5(E, a) = 9
S5(E, b) = 9
S5(E, c) = 8
S5(E, d) = 7
S5(E, e) = 6
S5(E, f) = 10

Stage 2: Select pairs which satisfy S > 6.

Client	Matched Set of Animals	Number of Matches
A	{b, d, e}	|A| = 3
B	{b}	|B| = 1
C	{a, b, c, e}	|C| = 4
D	{ }	|D| = 0
E	{a, b, c, d, f}	|E| = 5

Animal	Matched Set of Clients	Number of Matches
a	{C, E}	|a| = 2
b	{A, B, C, E}	|b| = 4
c	{E}	|c| = 3
d	{A, C, E}	|d| = 3
e	{A, C}	|e| = 2
f	{E}	|f| = 1

Stage 3: Generate a set of client-animal pairs as the output of this sub algorithm.

The matrix below shows the resulting client-animal pairs, where 1 stands for a valid pair (i.e., S>6) and 0 stands for an invalid pair (i.e., S<=6). Cells that have a value of 1 will be extracted in pairs in the form of (client_i, animal_i) and put into a set R as the output. The set R is then used as input for the next algorithm – Adoption Maximization.

Result:

Client\Animal	a	b	c	d	e	f
A	0	1	0	1	1	0
B	0	1	0	0	0	0
C	1	1	0	1	1	0
D	0	0	0	0	0	0
E	1	1	1	1	0	1

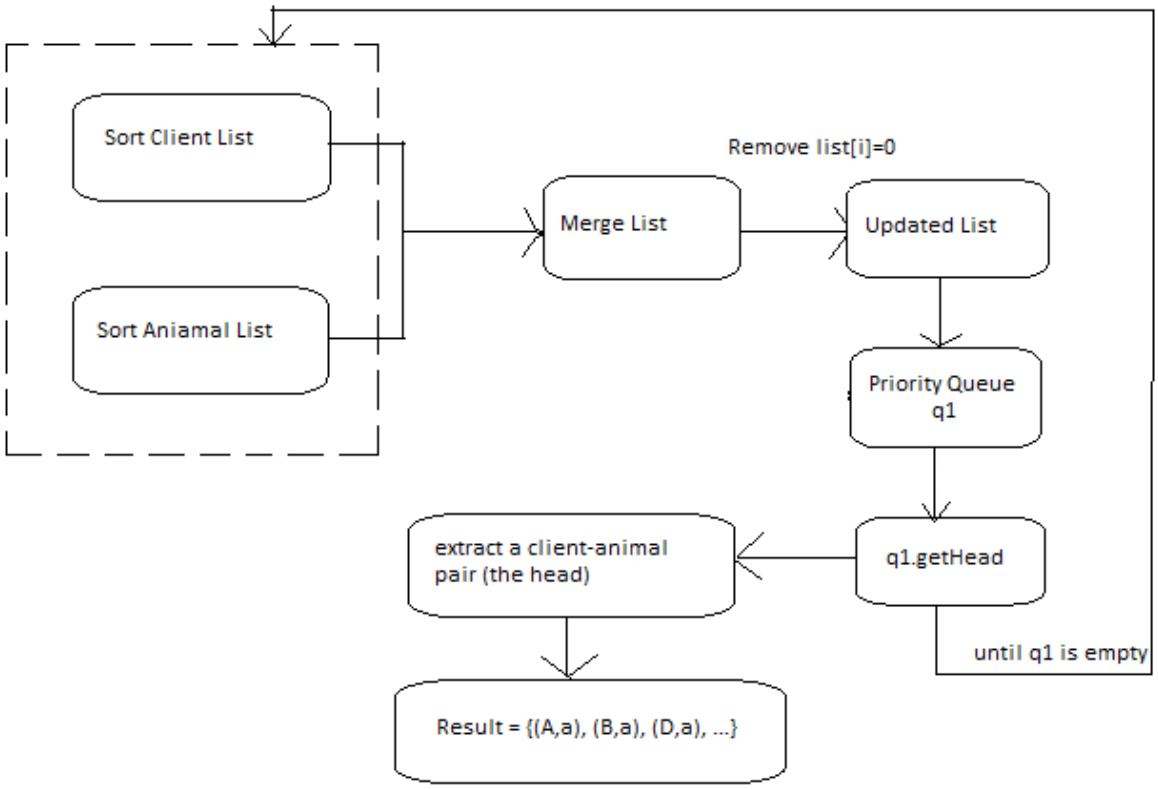
4. Computing Optimal Set of Matches

This section explains how the optimal set of matches are computed using the combination of Profile Matching sub algorithm and Adoption Maximization sub algorithm. First, the Adoption Maximization sub algorithm will be introduced and explained, then a detailed description will be provided stating how the ACM matching rules are used together with the Adoption Maximization sub algorithm to generate the final output of ACM algorithm.

4.1. Adoption Maximization Sub Algorithm

Adoption Maximization sub algorithm aims to take the best interests of all sheltered animals while producing relatively favourable matches to the clients. This sub algorithm can be referred to as the end algorithm. It takes the result from the previous algorithm (begin algorithm) and outputs a set of optimized client-animal matches in the end. The graph below illustrates the execution sequence of the Adoption Maximization sub algorithm.

Graph 2 – Flow of the Adoption Maximization Sub Algorithm



Stage 1: In the previous section (section 3-3.2), the cardinality of each list element in the result – number of matches, is computed during *stage 2*. Sort their cardinality from smallest to largest and put them into list_1.

For Clients				
|D|	|B|	|A|	|C|	|E|
0	1	3	4	5

Therefore, list_1 = [“D”, “B”, “A”, “C”, “E”].

Similarly, sort the animal list using the same principle, and put them into list_2.

For Animals					
|c|	|f|	|a|	|e|	|d|	|b|
1	1	2	2	3	4

Therefore, list_2 = [“c”, “f”, “a”, “e”, “d”, “b”].

Then, merge the two sorted lists into a single list list_3.

|D|	|c|	|f|	|B|	|a|	|e|	|d|	|A|	|C|	|b|	|E|
0	1	1	1	2	2	3	3	4	4	5

Therefore, list_3 = [“D”, “c”, “f”, “B”, “a”, “e”, “d”, “A”, “C”, “b”, “E”].

Stage 2: From list_3, delete list elements that hold a value of 0. This action is to remove clients/animals that have no valid matches from the list.

|c|	|f|	|B|	|a|	|e|	|d|	|A|	|C|	|b|	|E|
1	1	1	2	2	3	3	4	4	5

Therefore, updated list_3 = [“c”, “f”, “B”, “a”, “e”, “d”, “A”, “C”, “b”, “E”].

Stage 3: Build a priority queue based on list_3, where list elements that hold the smallest value have top priority. As a result, Animal c has the top priority, and then Animal f has the second priority, and Client B has the third priority.

Clients or Animals that have the top priorities are considered first for matching (FIFO). Therefore, the head of the priority queue is extracted as a pair of matches. After the head is popped from the priority queue, update all lists and queues that were previously stated.

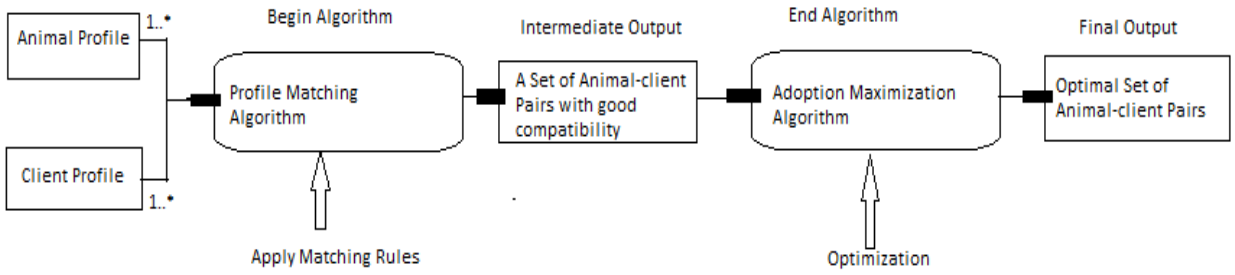
Stage 4: Keep repeating the operations of *Stage 2* and *Stage 3* until the priority queue is empty. Put all valid client-animal pairs generated by the previous matching algorithm into a set R.

Stage 5: Output set R as the final result of the ACM algorithm.

4.2. Combining Matching Rules and Optimization

As previously stated, the two sub algorithms work together to provide the full feature of the ACM algorithm. Profile Matching uses a unique set of rules to ensure the animal-client pairs are compatible and gives the basic matching result. Adoption Maximization optimizes the matching result from the previous algorithm and gives a set of recommended output matches to the users. The latter algorithm aims to benefit as many cuACS participants as possible, including human clients and sheltered animals.

The high-level block diagram below shows the flow of execution of the ACM algorithm. Combining matching rule together with optimization method enables the system to perform at full-fledged functionality:



The algorithm design at this stage is not yet finalized: modifications and updates could take place occasionally. More intelligent rules will possibly be added, and more thoughtful optimization will be introduced at a later stage.

5. Summary

The ACM algorithm is the core of the cuACS software system. Two sub algorithms – Profile Matching and Adoption Optimization, are established and utilized to produce the full functionality of ACM algorithm. With a well-designed Profile Matching algorithm, the cuACS will find a nice home for sheltered animals intelligently and fulfill the needs of the clients effectively. Once the Adoption maximization algorithm is added, the cuACS will be optimized even more such that the largest possible group of clients as well as the largest possible group of animals would benefit from the matching result of the cuACS software system.