ФЕДЕРАЛЬНОЕ АГЕНСТВО ЖЕЛЕЗНОДОРОЖНОГО ТАРНСПОРТА

Федерально государственное бюджетное образовательное учреждение
высшего образования
«Иркутский государственный университет путей и сообщения»
(ФГБОУ ВО ИрГУПС)

Факультет «Управление на транспорте и информационнные технологии»

Кафедра «Информационные системы и защита информации»

ОТЧЕТ ПО ПРАКТИКЕ

Производственная - по получению профессиональных умений и опыта профессиональной деятельности (технологическая)

ПП.430200.090404.000.ПЗ

Выполнил:
студент группы ПИм.1-16-1, Арляпов С.В.
Шифр: 1621345

«___»_____20__г.

Проверил:
ст. пр. Звонков И.В.
_____

«___»_____20__г.

Иркутск 2018

Содержание

Задание на практику **3**

Введение **4**

1 Теоретическая часть **5**

1.0.1 Группа формул Ψ 5

1.0.2 Группа формул Ф 6

2 Программная реализация **7**

3 Пример работы реализации **9**

Заключение **11**

Литература **12**

Задание на практику

Разработать модуляцию работы системы (группы) лифтов, используя нтеллуктуальный алгоритм обработки запросов.

В ходе практики должны быть освоены компетенции:

- способность анализировать профессиональную информацию, выделять в ней главное, структурировать, оформлять и представлять в виде аналитических обзоров с обоснованными выводами и рекомендациями
- способность проектировать вспомогательные и специализированные языки программирования и языки представления данных;
- способность проектировать распределенные информационные системы, их компоненты и протоколы их взаимодействия.

Введение

Автоматизированные системы управления активно приходят в повседневную жизнь человечества. Сначала, это были системы для управления производственным процессом на крупных предприятиях, теперь данные системы решают и бытовые задачи. Одной из таких задач является доставка человека с одного этажа на другой. Данная задача достаточно подробно описана в книге С.В. Васильева [1], там имеются абстрактная модель, логическая модель и логический вывод. С предложенным логическим выводом справилась бы система автоматического доказательства теорем А.А. Ларионова [2], но реализация всей системы лифтов на позитивно образованных формулах требует больших трудозатрат и будет носить чисто исследовательский характер.

Построить имитационную модель полностью на логическом выводе достаточно сложно, поэтому следует упростить задачу. А значит необходимо разбить модель на две части: систему взаимодействующих объектов и аппарат принятия решений. Таким образом, разработку обоих частей можно вести независимо дуг от друга. Поскольку при разработке имитационной модели можно сразу не иметь готовую логическую часть, а только ту, что будет содержать основные правила и реализовывать первый рассмотренный подход. Получив стабильную модель системы взаимодействия объектов, можно начать разработку и тестирование более сложных алгоритмов логического вывода. В дальнейшем можно будет усложнять имитационную модель, добавлять дополнительные условия и факторы, тем самым приближая модель к реальным условиям.

1 Теоретическая часть

Выстраивая логическую модель, получаем тройку (F, S, V), где F – это позитивно образованная формула (ПОФ), описывающая состояние лифта и принципы, по которым функционирует лифт, S – порядок ответов на запросы при логическом выводе, V – внешние воздействие, в данном случае имитация пассажиропотока. Так же следует отметить, что одними из основных будут предикаты связанные со временем: T(t) – момент времени t и N(t, t') – следующий за t момент времени t'.

Основными объектами в данной модели являются кабина Cab и человек Man. В момент времени t кабина имеет вид Cab(i, e, S, t), где i – идентификатор кабины, e – этаж, а S - маршрут кабины, список этажей. Человек имеет вид Man(e, d, τ, t), где e – этаж, d – целевой этаж, который добавляется в маршрут S в момент входа человека в кабину и d ≠ e, τ – длительность ожидания человеком кабины. Дистанцией же будет Dist(e, S, i, t, α), где α – это дистанция от кабины i на этаже e с маршрутом S, где произошёл вызов. И связь i кабины с вызовом с e этажа Conn(i, e).

В каждый момент времени t_0 принятия решения формула F имеет вид:

$$\exists A(t_0) \begin{cases} \forall T(t) \exists T(t'), N(t, t'), \\ \Phi \\ \Psi \end{cases}$$

$A(t_0)$ – коньюнкт, описывающий состояние системы в момент времени t_0. Если $A(t_0)$ содержит Man, то появление человека необходимо связать с вызовом определённой кабины. И группа формул Фпорождает все варианты связи и имитирует движение кабин совместно с формулой времени $\forall T(t) \exists T(t'), N(t, t')$ для некоторого количества тактов. А за счёт формул Ψпроисходит фильтрация некоторых вариантов.

Оставшиеся варианты оцениваются оцениваются и выбирается один из самых наилучших.

1.0.1. Группа формул Ψ

$$\forall Man(e, d, , t), N(t, t') \exists Dist(e, S_1, 1, t, \alpha_1), ...,$$

$$Dist(e, S_k, k, t, \alpha_k) \begin{cases} \exists Conn(1, e) \\ ... \\ \exists Conn(k, e), \\ \exists Man(e, d, +1, t') \end{cases}$$

– формула вычисляющая дистанцию до каждой кабины при новом появлении человека

$$\forall Cab(i, e, S, t), Conn(i, e'), N(t, t') \begin{cases} \forall e' < e \exists Cab(i, e-1, S/e', t') \\ \forall e < e' \exists Cab(i, e+1, S/e', t') \end{cases}$$

$$\forall Cab(i,e,S,t), S = S(e',S_1), N(t,t') \begin{cases} \forall e = e' \exists Cab(i, e-1, S_1, t') \\ \forall e' < e \exists Cab(i, e-1, S, t') \\ \forall e < e' \exists Cab(i, e+1, S, t') \end{cases}$$

$$\forall Cab(i,e,null,t), N(t,t') \exists Cab(i,e,null,t')$$

– формулы реализующие движение, где S/e является операцией вставки этажа в маршрут.

1.0.2. Группа формул Ф

Формулы из группы Фреализуют дополнительные ограничения, которые следует учитывать при построении логического вывода. В дальнейших работах они использоваться не будут. Но упомянуть о них крайне необходимо, так как они дают возможность данной модели быть более гибкой к различным ситуациям. А также их добавление в разрабатываемую систему не будет сложной задачей.

Например, вот формула, которая запрещает откладывать связывание вызова лифта человеком белее, чем на 4 такта:

$\forall Man(e,d,4,t) \exists False$

Это правили необходимо, в том случае если в модели будет возможна отсрочка принятия решения на вызов лифта.

2 Программная реализация

Однако, пусть реализация логического вывода на языке Prolog является целесообразной задачей, но для разделения моделируемой системы на логический блок и блок взаимодействия объектов необходима клиент-сервергая связка. А реализация сервера или клиента на языке Prolog не является его типовой задачей, что и касается реализации графической составляющей системы моделирования.

Таким образом более целесообразном будет оставить блок взаимодействия объектов реализованными на языку Prolog. Более того, правила описанные для модуляции системы будут использованы для построения решения логическим блоком.

Данный код иллюстрирует реализацию формулы реализующие обход возможных вариантов будущего, сбор статистики с каждого варианта и выбор наиболее подходящего варианта будущего по признаку. в данном случае интересующим признаком является среднее ожидание человеком кабины.

```
init_sim(SimPrefix) :-
        copy_var(SimPrefix, step),
        copy_var(SimPrefix, people_targets),
        copy_var(SimPrefix, people_floors),
        copy_var(SimPrefix, people_waiting),
        copy_var(SimPrefix, people_states),
        copy_var(SimPrefix, elevators_floors),
        nb_getval(n_elevators, NElev),
        copy_elev_list(SimPrefix, NElev).
simulate_loop_step :-
        logtrace('Do loop step'),
        manage_people,
        manage_elevators.
simulate_loop(SimPrefix) :-
        nb_setval(current_sim, SimPrefix),
        sim_getval(SimPrefix, step, Step),
        nb_getval(n_steps, Steps),
        Step >= Steps,
        swritef(SimLog, 'Simulation \'%t\' is finished', [SimPrefix]),
        logtrace(SimLog),
        show_stat.
simulate_loop(SimPrefix) :-
        nb_setval(current_sim, SimPrefix),
        var_getvalue(step, Step),
        nb_getval(n_steps, Steps),
        Step < Steps,
        logtrace('Start step'),
        simulate_loop_step,
        logtrace('Finish step'),
        Next is Step + 1,
        var_setvalue(step, Next),
        simulate_loop(SimPrefix).
simulate :-
        (nb_current(current_sim, CurrentPrefix) ->
                SimPrefix = CurrentPrefix
        ;
                SimPrefix = 'r_'
        ),
        init_sim(SimPrefix),
        simulate_loop(SimPrefix).
do_simulate(Floor, Elev, H) :-
        nb_getval(current_sim, SimPrefix),
        append2map(Elev, Floor),
        manage_elevators,
        var_getvalue(step, Step),
        Next is Step + 1,
        var_setvalue(step, Next),
        simulate_loop(SimPrefix),
        sim_getval(SimPrefix, people_waiting, PeopleWaitingList),
        sum_list(PeopleWaitingList, H),
```

```
        swritef(SimLog, 'Waiting sum is \'%t\'', [H]),
        logdebug(SimLog).
simulate(Floor, Elev, H) :-
        (nb_current(current_sim, CurrentPrefix) ->
                atom_concat(CurrentPrefix, Elev, SimPrefix),
                atom_concat(SimPrefix, '_', Prefix),
                init_sim(Prefix),
                nb_setval(current_sim, Prefix),
                do_simulate(Floor, Elev, H),
                nb_setval(current_sim, CurrentPrefix)
        ;
                atom_concat('r_', Elev, SimPrefix),
                atom_concat(SimPrefix, '_', Prefix),
                init_sim(Prefix),
                nb_setval(current_sim, Prefix),
                do_simulate(Floor, Elev, H),
                nb_delete(current_sim)
        ).
calculating(_, Elev, []) :-
        nb_getval(n_elevators, NElev),
        Elev >= NElev.
calculating(Floor, Elev, [H | T]) :-
        nb_getval(n_elevators, NElev),
        Elev < NElev,
        simulate(Floor, Elev, H),
        Next is Elev + 1,
        calculating(Floor, Next, T).
calculate(Floor, ResList) :-
        calculating(Floor, 0, ResList).
do_elev_call(Floor, Elev) :-
        calculate(Floor, ResList),
        min_list(ResList, Min),
        nth0(Elev, ResList, Min).
```

Благодаря функционалу SWI-Prolog предикаты отражающие состояние системы в момент вызова кабины можно будет указывать в правилах только при необходимости.

3 Пример работы реализации

Данный пример иллюстрирует работу группы лифтов, где их количество равно 2, в здании с 5 этажами. В ходе работы системы появится два человека в моменты времени t_2 и t_4 на этажах e_1 и e_0, и у каждого человека целью будет четвёртый этаж d_4.

Изначально первая кабина находится на первом этаже e_0, а вторя на втором e_1. Ниже приведён лог показывающий работу системы:

```
24 May 2018 06:49:49 [71:null] TRACE Start modulation
24 May 2018 06:49:49 [71:null] WARNING Var 'Step' has null value
24 May 2018 06:49:49 [71:null] INFO Set first step '0'
24 May 2018 06:49:49 [71:0] TRACE Start step
24 May 2018 06:49:49 [71:0] TRACE Finish step
24 May 2018 06:49:49 [71:1] TRACE Start step
24 May 2018 06:49:49 [71:1] TRACE Finish step
24 May 2018 06:49:49 [71:2] TRACE Start step
24 May 2018 06:49:49 [71:2] INFO A man appears with id '0' on floor with id '1'
24 May 2018 06:49:49 [71:2] DEBUG Change list 'people_states' with id '0' res:'[1,0]'
24 May 2018 06:49:49 [71:2] TRACE Putting floor to map
24 May 2018 06:49:49 [71:2] DEBUG Current distances are '[1,0]'
24 May 2018 06:49:49 [71:2] DEBUG Current min dist '0' with id '1'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'step' into 'r_0_step' with val '2'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'people_targets' into 'r_0_people_targets' with
    val '[4,4]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'people_floors' into 'r_0_people_floors' with val
    '[1,0]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'people_waiting' into 'r_0_people_waiting' with
    val '[0,0]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'people_states' into 'r_0_people_states' with val
    '[1,0]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'elevators_floors' into 'r_0_elevators_floors'
    with val '[0,1]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'elev_rmap_1' into 'r_0_elev_rmap_1' with val '[]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'elev_people_1' into 'r_0_elev_people_1' with val
    '[]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'elev_rmap_0' into 'r_0_elev_rmap_0' with val '[]'
24 May 2018 06:49:49 [71:2] TRACE Copy var 'elev_people_0' into 'r_0_elev_people_0' with val
    '[]'
24 May 2018 06:49:49 [71:2] [r_0_:2] DEBUG Append to road map '[1,-1]' floor '1'
24 May 2018 06:49:49 [71:2] [r_0_:2] DEBUG Elev floors '[0,1]'
24 May 2018 06:49:49 [71:2] [r_0_:2] TRACE moving_elevators NElev '2' Id '0'
24 May 2018 06:49:49 [71:2] [r_0_:2] DEBUG Elevator on '0' has goal '1'
24 May 2018 06:49:49 [71:2] [r_0_:2] TRACE Change ElevPos '0' Id '0'
24 May 2018 06:49:49 [71:2] [r_0_:2] DEBUG Move elevator '0' from '0' to '1'
24 May 2018 06:49:49 [71:2] [r_0_:2] DEBUG Current elev '0' road map '[1,-1]' people '[]'
24 May 2018 06:49:49 [71:2] [r_0_:2] TRACE moving_elevators NElev '2' Id '1'
24 May 2018 06:49:49 [71:2] [r_0_:2] DEBUG Current elev '1' road map '[]' people '[]'
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE Start step
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE Do loop step
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG A man with id '0' has been waiting for '0'
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Change list 'people_waiting' with id '0' res:'
    [1,0]'
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Elev floors '[1,1]'
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE moving_elevators NElev '2' Id '0'
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Elevator on '1' has goal '1'
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Elevator '0' has reached '1'
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE Manage people
24 May 2018 06:49:49 [71:2] [r_0_:3] INFO Going out people '[]' from elev '0'
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE Trace getting_people_in '2'
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE Trace getting_people_in '1'
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Change list 'people_states' with id '0' res:'[2,0]
    '
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Append to road map '[-1,4,-1]' floor '4'
24 May 2018 06:49:49 [71:2] [r_0_:3] INFO Coming people '[0]' to elev '0'
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Current elev '0' road map '[-1]' people '[0]'
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE moving_elevators NElev '2' Id '1'
24 May 2018 06:49:49 [71:2] [r_0_:3] DEBUG Current elev '1' road map '[]' people '[]'
24 May 2018 06:49:49 [71:2] [r_0_:3] TRACE Finish step
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Start step
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Do loop step
24 May 2018 06:49:49 [71:2] [r_0_:4] INFO A man appears with id '1' on floor with id '0'
24 May 2018 06:49:49 [71:2] [r_0_:4] DEBUG Change list 'people_states' with id '1' res:'[2,1]
```

```
    '
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Putting floor to map
24 May 2018 06:49:49 [71:2] [r_0_:4] DEBUG Current distances are '[9,1]'
24 May 2018 06:49:49 [71:2] [r_0_:4] DEBUG Current min dist '1' with id '1'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'step' into 'r_0_0_step' with val '4'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'people_targets' into '
    r_0_0_people_targets' with val '[4,4]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'people_floors' into 'r_0_0_people_floors
    ' with val '[1,0]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'people_waiting' into '
    r_0_0_people_waiting' with val '[1,0]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'people_states' into 'r_0_0_people_states
    ' with val '[2,1]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'elevators_floors' into '
    r_0_0_elevators_floors' with val '[1,1]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'elev_rmap_1' into 'r_0_0_elev_rmap_1'
    with val '[]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'elev_people_1' into 'r_0_0_elev_people_1
    ' with val '[]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'elev_rmap_0' into 'r_0_0_elev_rmap_0'
    with val '[-1,4,-1]'
24 May 2018 06:49:49 [71:2] [r_0_:4] TRACE Copy var 'elev_people_0' into 'r_0_0_elev_people_0
    ' with val '[0]'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] DEBUG Append to road map '[-1,0,-1,4,-1]' floor '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] DEBUG Elev floors '[1,1]'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] TRACE moving_elevators NElev '2' Id '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] DEBUG Elevator on '1' has goal '-1'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] TRACE Elevator '0' is closing the door
24 May 2018 06:49:49 [71:2] [r_0_0_:4] TRACE Change ElevPos '1' Id '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] DEBUG Move elevator '0' from '1' to '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] DEBUG Current elev '0' road map '[0,-1,4,-1]' people '
    [0]'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] TRACE moving_elevators NElev '2' Id '1'
24 May 2018 06:49:49 [71:2] [r_0_0_:4] DEBUG Current elev '1' road map '[]' people '[]'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] TRACE Start step
24 May 2018 06:49:49 [71:2] [r_0_0_:5] TRACE Do loop step
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG A man with id '1' has been waiting for '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Change list 'people_waiting' with id '1' res:'
    [1,1]'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Elev floors '[0,1]'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] TRACE moving_elevators NElev '2' Id '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Elevator on '0' has goal '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Elevator '0' has reached '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] TRACE Manage people
24 May 2018 06:49:49 [71:2] [r_0_0_:5] INFO Going out people '[]' from elev '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] TRACE Trace getting_people_in '2'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Change list 'people_states' with id '1' res:'
    [2,2]'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Road map '[-1,4,-1]' has such floor '4'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] TRACE Trace getting_people_in '1'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] INFO Coming people '[1]' to elev '0'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Current elev '0' road map '[-1,4,-1]' people '
    [0,1]'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] TRACE moving_elevators NElev '2' Id '1'
24 May 2018 06:49:49 [71:2] [r_0_0_:5] DEBUG Current elev '1' road map '[]' people '[]'
```

Изучив выше изложенный журнал, можно увидеть, что происходит построение дерева формулы и её обход. Для того чтобы было нагляднее следует прокомментировать строку лога.

Первые четыре столбца в логе - это реальное время журналирования момента модуляции. Следующим столбцом идёт связка двух чисел, первое число - это номер процесса, он необходим для идентификации сессии, а второе число показывает момент времени модуляции. Ещё одним столбцом является связка строки и числа, число - это так же момент времени в данной ветки формулы, А строка отражает индекс чанка формулы в момент вывода, r означает корень выводимой формулы, а дальше серез нижние подчёркивание перечислены индексы кабин, которые участвуют в логическом выводе в данные момент.

Заключение

В результате прохождения практики была разработана программная реализация на языке Prolog. Данная реализация модулирует процесс работы и управления группы лифтов.

На данном этапе реализован интеллектуальный алгоритм управления, следующим шагом будет разработка вероятностной составляющей вариантов будущего.

А также получены практические навыки в системах с параллельной обработкой данных и высокопроизводительные системы, и их компоненты.

Литература

[1] С. Н. Васильев. Интеллектуальное управление динамическими системами / С. Н. Васильев, А. К. Жерлов, Е. А. Федосов, Б. Е. Федунов - М.. Физико-математическая литература, 2000. - 352 с.

[2] А. А. Ларионов. Программные технологии для эффективного поиска логического вывода в исчислении позитивно-образованных формул / А. А. Ларионов, Е. А. Черкашин – Иркутск : Изд-во ИГУ, 2013. – 104 с.