УДК 004.8+519.711

ИНТЕЛЛЕКТУАЛЬНОЕ УПРАВЛЕНИЕ: ОБЗОР

© В. И. Донской

Таврический национальный университет им. В. И. Вернадского
пр-т Вернадского, 4, Симферополь, Республика Крым, Россия
e-mail: *donskoy@tnu.crimea.ua*

Intelligent control: review.

Donskoy V. I.

Abstract. The paper presents a review of the intellectual control systems: foundation, applications, program agents. The intellectual control is the most interesting application domain of theoretical cybernetics in the modern information systems. Here all developed receptions, methods, various parts of theory are going together: recognition of the image-states, concept formation, identification, optimization based on incomplete information, self-organizing automata theory. Practically all methods of operations research are joined here.

Area of practical application of methods of intellectual control is extraordinarily wide and includes: programs-agents for the difficult (especially functioning in the network environment) information systems; agents for the production workshop management; intellectual programs for decision-making in the regional systems; control programs in robotics; programs-researchers for automatic spaces vehicles and great deal other.

The dynamic intellectual systems, which able to rebuild itself as far as they are functioning in an environment being surrounding of controlled object, are most difficult. To realize such systems the design of mechanism of information "forgetting" is required for information which becomes in a time irrelevant.

Prime directions of development of intellectual control theory appear: development of specific models of memory for agents, algorithmic knowledge bases for agents, models of dynamic automatic state classification of the controlled objects and environment, automatic change of structures of mathematical models and even automatic changing of model in-use by an agent.

Введение

В теории управления выделяют как основные элементы — *объект управления, внешнюю среду*, в которой функционирует этот объект, *систему управления*, которая на основе информации о состояниях внешней среды и объекта управления оказывает на него *управляющее воздействие*, а также *критерии качества управления и ограничения*, носящие ресурсный или технический характер.

Математическая теория управления абстрагируется от физических свойств перечисленных элементов и "работает" с математическими моделями управляемых

объектов, среды, систем управления и управляющих воздействий. Последние описываются информацией, представленной в той или иной форме — текстов, чисел, функций и пр.

Традиционное, классическое управление предполагает разработку моделей объектов и управляющих систем специалистом-математиком на основе заданных свойств перечисленных выше основных элементов.

Гибкое производственное управление предполагает ситуативную перестройку критериев и управляющих воздействий. В качестве примера реализации такого управления можно рассмотреть систему, основанную на динамическом отборе подходящих оперативных критериев и стратегий производственного расписания, которая представлена в работе [24]. Методология управления данной системы основана на двухуровневом механизме принятия решений. Первый уровень предназначен для выбора доминирующего критерия управления и релевантного управления расписанием производственного процесса при помощи алгоритма, основанного на правилах. На втором уровне в результате упреждающего многопроходного имитационного моделирования определяется расписание производственного процесса, которое позволяет как можно более улучшить значение критерия качества, выбранного на первом уровне.

Можно отметить тенденцию включения методов индуктивного обучения в инструментарий проектирования систем гибкого производственного управления [3, 14, 16, 25].

Обучаемое управление предполагает, что система управления обладает вычислительными возможностями, позволяющими извлекать и обобщать знания с целью порождения математической модели управляемой системы и модифицировать свое собственное управляющее воздействие на основе этих знаний [8].

Интеллектуальное управление характеризуется эмуляцией методов управления, свойственных человеку: предполагается использование адаптации, обучения, планирования в условиях неопределенности и обработки больших объемов разнотипных данных [1, 29].

Обучение является важным и определяющим свойством интеллектуального управления. Именно способность к обучению обеспечивает автономность поведения управляющей системы, обеспечивает её адаптивную самоорганизацию при изменении свойств объекта управления и/или внешней среды. Обучение может рассматриваться как процесс, посредством которого система управления может модифицировать свои функции для повышения качества функционирования управляемого объекта на основе извлечения релевантной информации (знаний).

В работе [2] Панос Антсаклис объединяет два приведенных выше термина в один, вводя "*Интеллектуальное управление с обучением*" (*Intelligent Learning Control*), что фактически не привносит ничего нового в рассматриваемую парадигму, но может быть оправдано центральной ролью обучения в выработке модификаций моделей объекта управления и функций управляющей системы. П.Антсаклис приводит следующую краткую классификацию направлений применения машинного обучения в интеллектуальном управлении:

– обучение, обобщающее знания об управляемом объекте (I);
– обучение, обобщающее знания об окружающей среде (II);
– обучение, обобщающее знания о самой управляющей системе (контроллере) с целью его модификации (III);
– *обучение, в результате которого синтезируется новая целевая функция и ограничения, определяющие возможности и качество функционирования объекта управления* (IV).

Несложно понять, что пункт (IV), вообще говоря, включает как необходимые элементы все остальные перечисленные выше пункты и является центральным в математической постановке задачи автоматического интеллектуального управления на основе синтеза функций цели и текущих ограничений методами машинного обучения.

Уместно упомянуть *адаптивное автоматическое управление*, под которым традиционно понимается параметрическая адаптация систем, как правило, с обратной связью и чаще всего — с целью повышения устойчивости (помехоустойчивости). Важнейшей особенностью интеллектуального управления, отличающей его от адаптивного управления, является возможность изменения структуры управления, применяемых моделей и даже принципов управления, а не только параметров. Хотя можно заметить, что адаптивное управление используется, в частности, при обучении нейронных сетей зафиксированной структуры путем модификации их параметров на основе наблюдаемой информации.

Если рассмотреть следующую упрощенную формулу: "интеллектуальное управление = теория управления + искусственный интеллект", то возникнет ощущение, что в ней чего-то не хватает. И любой опытный разработчик систем интеллектуального управления заметит: нужно обязательно добавить в приведенную формулу исследование операций, поскольку теория поиска на графах, теория очередей, теория игр и в целом теория оптимального выбора являются важнейшим инструментарием при разработке систем интеллектуального управления [12].

Сам процесс такой разработки обязательно будет являться многоуровневым и итерационным, что удачно проиллюстрировано схемой в работе [12] (Рис. 1).

Важную роль играет специфика процесса разработки, связанная со следующими основными свойствами выбираемых математических моделей:

– используются логические, многозначно-дискретные или непрерывные переменные;
– используется обучение по прецедентам (CBL — *Case Based Learning*), путем извлечения знаний (KBL — *Knowledge Base Learning*), гибридное обучение — *Hybrid Learning*);
– используется только параметрическая или также и структурная оптимизация обучаемых моделей управления;

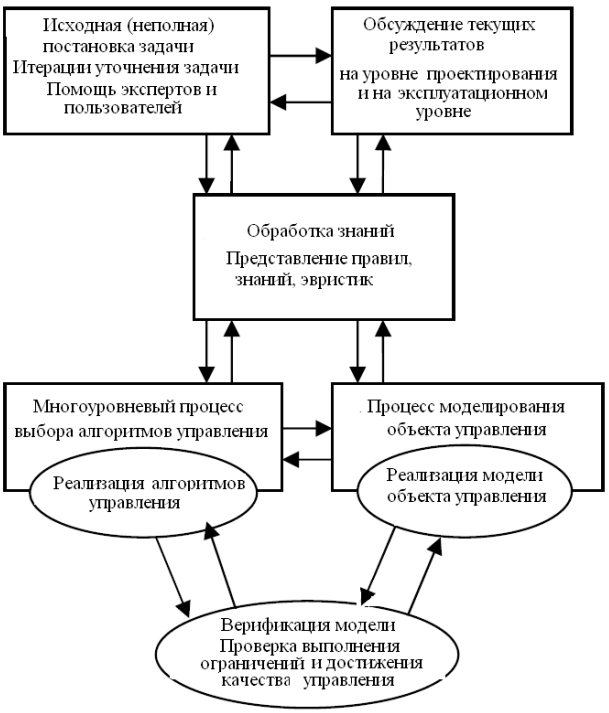

Рис. 1. *Схематическое представление процесса разработки систем интеллектуального управления* [12]

– реализуется ли возможность применения пополняемой и модифицируемой базы алгоритмов управления;
– создается ли специальная программная среда для моделирования и верификации создаваемой системы интеллектуального управления.

Приведенные особенности не исчерпывают всего многообразия факторов и свойств, которые сопутствуют процессам проектирования систем интеллектуального управления.

1. Опыт создания систем интеллектуального управления

В работе [26] представлен *интерактивный метод построения контроллера динамической системы на основе комбинирования извлечения знаний оператора, работавшего с этой системой, и машинного обучения.* Метод применялся для **имитации управления самолетом в тренажере полета**. Программа извлечения знаний генерировала правила, фиксируя действия пилота во время "полета". Схема управления приведена на рис. 2.

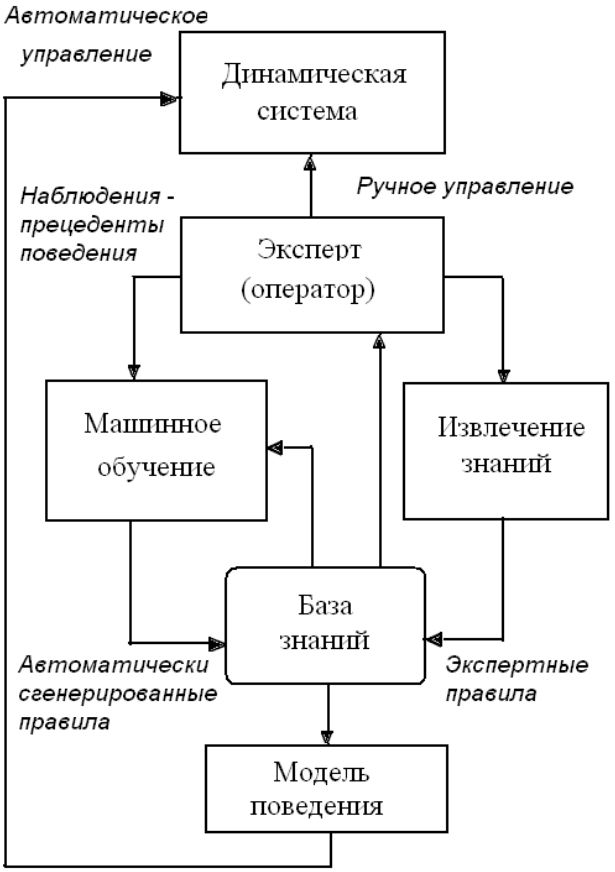

Рис. 2. Гибридная система интеллектуального управления, использующая прецеденты и экспертные знания – продукции [26].

В работе [7] предложен **подход к построению гибкого интеллектуального управления производственным цехом с множественной параллельной обработкой изделий**, основанный на применении решающих деревьев. Авторы этой работы использовали оригинальный четырехстадийный алгоритм синтеза решающего дерева, указав его преимущества перед алгоритмом $C5.0$ — улучшенной версией $C4.5$ [18]. Основные решения управляющей системы обеспечивают выбор пар $\langle\langle$операция-станок$\rangle\rangle$ и последовательности работ, назначенных каждому станку. Решающее дерево используется для выбора приоритетных правил, подходящих для определенных состояний системы, и затем — для выдачи требований в имитационную модель. Многократные имитации дают возможность уточнения управляющих решений.

Метод тестировался на учебном примере для цеха с реконфигурируемым оборудованием и продемонстрировал способность минимизировать общее время работ.

В статье [23] описана **система интеллектуального управления процессом сжигания антрацита**. Система получает информацию от датчиков, цифровое изображение пламени и автоматически определяет управляющую стратегию для процесса горения. Испытания на реальном энергетическом предприятии продемонстрировали способность системы обеспечивать увеличение эффективности и уменьшение эмиссии газов.

В работе [28] описана **интеллектуальная система управления расписанием производственной обработки заданий**, способная вырабатывать в реальном времени решения, согласованные с жесткими производственными требованиями. Система автоматически извлекает близкое к оптимальным подмножество аттрибутов-признаков. В частности, *использовались следующие атрибуты-признаки*:

Nj — число работ в системе;

MeUM — среднее число используемых машин (станков);

SdUM — среднеквадратичное отклонение числа используемых машин;

MeUL — среднее число загрузок/разгрузок;

MeUB — среднее число используемых буферных устройств;

MeUA — среднее число используемых автоматически управляемых тележек;

MiOT — минимальное время ожидания выполнения работы;

MAOT — максимальное время ожидания выполнения работы;

MeOT — среднее время ожидания выполнения работы;

SdOT — стандартное отклонение времени ожидания выполнения работы;

MiPT — минимальное общее время обработки в системе;

MAPT — минимальное общее время обработки в системе;

MePT — среднее общее время обработки в системе;
SdPT — стандартное отклонение общего времени обработки в системе;
MiRT — минимальный простой;
MaRT — максимальный простой;
MeRT — средний простой;
SdRT — среднее квадратичное отклонение простоев;
MiST — минимальное время ожидания;
MeST — среднее время ожидания;
SdST — среднее квадратичное отклонение времени ожидания;
MaTA — максимальное отставание;
Meta — среднее отставание;
SdTA — среднее квадратичное отклонение отставания;
MAWL — максимальная загрузка;
ToWL — общая загрузка;
MeSO — среднее время пребывания (заявки) в системе;
SdSO — среднее квадратичное отклонение времени пребывания в системе;
MeTD — среднее время отставания;
SdTD — среднее квадратичное отклонение времени отставания.

Атрибуты-признаки использовались для описания знаний, наполняющих соответствующую базу знаний. Предлагаемый подход интегрирует генетические алгоритмы и решающие деревья. Генетический алгоритм отыскивает подмножества признаков. Решающие деревья (был использован алгоритм $C4.5$ [18]) обучаются отбору правил управления производственным процессом.

Примеры управляющих правил:
FIFO — запускать работу в соответствии с правилом FIFO (*First In First Out*);
SPT — запускать работу с кратчайшим требуемым временем обработки;
SIO — запускать работу с кратчайшим временем, оставшимся до нормативного срока исполнения;
SRPT — запускать работу с кратчайшим остатком времени исполнения;
CR — запускать работу с минимальным отношением остатка времени до нормативного срока исполнения к оставшемуся требуемому времени исполнения работы.

Применение системы показало ее способность улучшать следующие *критерии производительности управляемого объекта*:
TP — пропускную способность;

MF — среднее время выполнения задания;
NT — число запаздываний.

Работа [15] посвящена проектированию **дискретного предсказывающего контроллера с бинарными управляющими действиями, использующего метод ближайшего соседа**. Применяемый kNN метод модифицирован: иногда выбирается не ближайший элемент данных, а другой, достаточно близкий, — с некоторой вероятностью. Такой метод называют локально чуствительным хешированием *locality sensitive hashing* (LSH) *method*. kNN, используемый совместно с LSH в работе [15] называется аппроксимирующим методом — *approximate nearest neighbor* ANN. Контроллер использовался для управления высоковольтным электромотором.

Достаточно интересным и эксклюзивным является **использование интеллектуального управления музыкальным синтезатором в реальном времени** при "живом" исполнении произведений [27]. Музыканты часто изменяют параметры синтезаторов в процессе исполнения, чтобы достичь большей выразительности звучания. Машинное обучение позволяет обеспечить автоматическое регулирование параметров на основе анализа примеров исполнения и обеспечить выразительный звук с желательным эстетическим эффектом. В статье [27] представлены результаты продолжающегося исследования возможностей *нейронных сетей долговременной памяти для коротких термов* — LSTM (*Long Short-Term Memory NNs*).

Сети LSTM содержат ячейки памяти, которые сохраняют значения данных, проходящих через сеть, а также три типа специализированных узлов активации, называемых вычислительными элементами — гейтами (*gates*). Группы ячеек памяти и связанные с ними вычислительные элементы объединяются в *блоки*.

Входные вычислительные элементы управляют доступом к ячейкам памяти, осуществляя нужное масштабирование; *выходные* вычислительные элементы управляют доступом к чтению из ячеек памяти; вычислительные элементы *забывания* дают возможность сброса содержимого ячеек памяти на основе оценивания текущих значений.

Вычислительные элементы — гейты имеют взвешенные связи на входах и рекурентные связи с другими вычислительными элементами и элементами памяти.

В процессе обучения вычислительные элементы получают "навык" открываться и закрываться таким образом, чтобы ячейки памяти могли запоминать и накапливать значения, удерживать их в течение произвольных периодов времени, и использовать значения ячеек, чтобы воздействовать на работу сети нужным образом.

На рис. 3 приведена структура LSTM блока с одной ячейкой памяти.

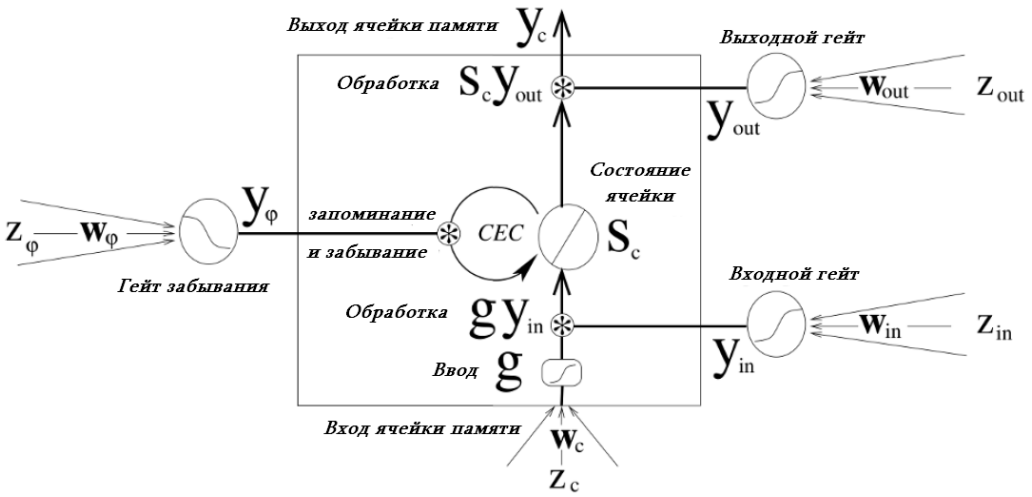

Рис. 3. *Блок LSTM с единственной ячейкой памяти из работы* [11]. *Активации гейтов и ячейки памяти вычисляются как взвешенные суммы входных связей при помощи функций активации подобно тому, как это происходит в обычных искусственных нейронных сетях. Вход в ячейку памяти управляется активацией входного гейта, выход — активацией выходного гейта, а состояние ячейки управляется активацией гейта забывания* [27].

2. Математические методы, применяемые для создания систем интеллектуального управления

Применяемые в задачах интеллектуального управления математические методы определяются, в основном, природой объекта управления и среды — их особенностями и *сложностью*. Сложность управляемых объектов может быть столь высокой, что прямое применение некоторых математических методов становится практически невозможным. Если к тому же имеет место неопределенность исходной информации, доступной для выработки управляющих воздействий, то потребуются математические методы, допускающие такую неопределенность, и это, прежде всего (но не только), — методы искусственного интеллекта: обучение, извлечение знаний, распознавание, предсказание. При этом во всех случаях в той или иной форме понадобится применение (возможно, специфических) методов оптимизации.

При условии линейности объекта управления применяются *методы обучения на основе H_∞ управления* — метода теории управления для синтеза оптимальных регуляторов.

(Обозначение H_∞ в комплексном анализе связано с нормой в пространстве Харди; p — норма Харди функции f имеет вид

$$\sup_{0<r<1}\left(\frac{1}{2\pi}\int_0^{2\pi}|f(re^{i\theta})|^p d\theta\right)^p.$$

Как и в случае пространств L^p, эта норма обобщается на случай $p=\infty$ и принимает вид $\sup\limits_{z:|z|<1}|f(z)|$).

В работе [31] описан **синтез обучаемого контроллера с обратной связью для линейных объектов управления с ограниченной по норме параметрической неопределенностью**. Предложен итерационный обучающий алгоритм для систем с обратной связью. Проблема синтеза итерационного обучаемого управления переформулирована как γ-субоптимальная H_∞ проблема управления на основе дробно-линейного преобразования. Достаточное условие сходимости итерационного процесса обучения системы управления представлено в терминах линейных матричных неравенств.

Итерационная обучаемая система управления с обратной связью, действующей в реальном времени, показана на рис. 2.

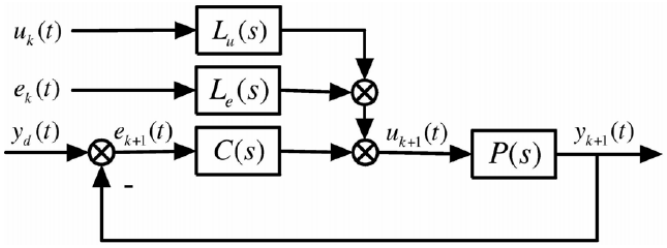

Рис. 4. Управление с обучением из работы [31]

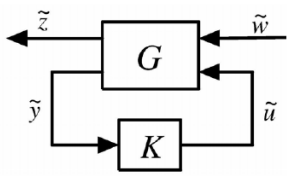

Рис. 5. Обычное управление с обратной связью

Использованы следующие обозначения:

$y_d(t)$ — желаемая траектория выхода на конечном интервале $t\in[0,T]$;

t — переменная временной области; s — переменная частотной области;

$y_k(t)$ — выход системы;

$u_k(t)$ — вход управления;

$e_k(t)$ — ошибка на k-той итерации;

$P(s)$ — управляемый объект;

$L_u(s)$ — обучаемый контроллер;

$L_e(s)$ — обучаемый контроллер;

$C(s)$ — контроллер обратной связи.

Алгоритм обучения с обратной связью представляется выражением

$$U_{k+1}(s) = L_u(s)U_k(s) + L_e(s)E_k(s) + C(s)E_{k+1}(s).$$

Для заданного обучаемого контроллера $L_u(s)$ задача построения обучаемого управления сводится к нахождению $C(s)$ и $L_e(s)$ путем решения субоптимальной H_∞ проблемы синтеза:

$$\left\|(I + P(s)C(s))^{-1}(L_u(s) - P(s)L_e(s))\right\|_\infty = \gamma < 1.$$

Далее, путем матричных преобразований, субоптимальная проблема синтеза сводится к решению традиционной задачи синтеза управления с обратной связью (рис. 3).

В работе [21] описан подход к **управлению для марковского процесса принятия решений** (МППР), результирующие траектории которого описываются средствами линейной темпоральной логики (ЛТЛ). Реализация МППР осуществляется с использованием детерминированного автомата Рабина, сгенерированного с учетом требуемых свойств ЛТЛ. Функция поощрения определяется степенью соответствия выходной траектории управления условиям, определяемым автоматом Рабина. Такая конструкция позволяет применить метод машинного обучения для синтеза спецификации ЛТЛ даже в том случае, когда переходные вероятности марковского процесса априорно неизвестны.

ЛТЛ позволяет выражать такие свойства систем, как устойчивость, наблюдаемость, живучесть, безопасность.

ЛТЛ формулы строятся из атомарных предложений $\omega \in \Pi$, которые являются описаниями состояний системы и оцениваются значениями “истина” или “ложь”. Пропозициональная формула ϕ строится из атомарных предложений, булевых операций и темпоральных операторов, в частности, таких [21]:

G_ϕ — ϕ истинна во всех будущих моментах времени;
F_ϕ — ϕ истинна в некоторых будущих моментах времени;
X_ϕ — ϕ истинна в следующем моменте времени;
$\phi_1 U \phi_2$ — ϕ_1 истинна, пока истинна ϕ_2.

Использование ЛТЛ позволяет описывать, например, такие свойства как наблюдаемость — GF_ϕ и живучесть — FG_ϕ.

Детерминированным автоматом Рабина (ДРА) называется [21] пятерка $\mathscr{R} = \langle Q, \Sigma, \delta, q_0, F\rangle$, где Q — множество состояний, Σ — входной алфавит, $\delta : Q \times \Sigma \to Q$ — функция переходов; q_0 — начальное состояние; $F = \{(G_1, B_1), \ldots, (G_{n_F}), (B_{n_F})\}$, $G_i, B_i \in Q$, $i = 1, \ldots, n_F$, — множество допускающих условий.

Выходом автомата Рабина является бесконечная последовательность состояний, начинающаяся с q_0 и определяемая функцией переходов δ. Для произвольной выходной последовательности r ДРА $inf(r)$ обозначает множество состояний, встречаемых бесконечно много раз в последовательности r. Последовательность $q_0, q_1, \ldots$ *принимается*, если найдется такое $i \in \{1, \ldots, n_F\}$, что $inf(r) \cap G_i \neq \varnothing$ и $inf(r) \cap B_i = \varnothing$.

Для любой ЛТЛ формулы ϕ над Π может быть сконструирован детерминированный автомат Рабина со входным алфавитом $\Sigma = 2^{\Pi}$ такой, что он будет принимать те и только те слова над Π, которые удовлетворяют формуле ϕ. Такой "принимающий" ϕ автомат обозначается $\mathscr{R}_\phi$.

Помеченным марковским процессом принятия решений называется шестерка $\mathscr{M} = \langle S, \mathfrak{A}, P, s_0, \Pi, L\rangle$, где S — конечное число состояний процесса; A – конечное число управляющих воздействий; $\mathfrak{A} : S \to 2^A$ — отображение состояний во множество действий; P — функция определения переходных вероятностей, $P : S \times A \times S \to [0, 1]$; $s_0 \in S$ — начальное состояние; Π — множество атомарных предложений; $L : S \to 2^{\Pi}$ — функция пометок, размечающая состояния атомарными предложениями.

Стратегией для МППР $\mathscr{M}$ называется функция $\pi : S^+ \to A$, где S^+ — множество всех конечных последовательностей состояний, такая что $\pi(s_0 s_1 \ldots s_n) \in \mathfrak{A}(s_n)$. Стратегия π для МППР $\mathscr{M}$ индуцирует марковскую цепь $\mathscr{M}_\pi$, которая описывается последовательностями $s_0 s_1 \ldots s_i$ смены состояний согласно вероятностям $P(s_i, a, s_{i+1})$, $a \in A$.

Подход, предлагаемый в рассматриваемой работе основан на нахождении такой стратегии, которая *максимизирует ожидаемую полезность* дополнительного МППР, построенного по исходному МППР и желаемой ЛТЛ спецификации. ЛТЛ спецификация преобразуется в детерминированный автомат Рабина ДРА, затем конструируется производный МППР такой, что его состояния дополняют состояния этого автомата. Предполагается 1) нахождение переходных вероятностей (вероятностей смены состояний и выдачи управляющего воздействия) путем обучения с поощрением и 2) оптимизация ожидаемой полезности. В работе приведен соответствующий алгоритм обучения.

Доказана теорема о реализуемости требований ЛТЛ спецификаций.

Для иллюстрации излагаемого подхода авторы работы [21] приводят пример интеллектуального агента, управляющего клеточной игрой 5×5, в которой требуется посещать области, помеченные α и β, избегая посещения области, помеченной γ. ЛТЛ спецификация представляется формулой

$$GF\alpha \wedge GF\beta \wedge G\neg\gamma.$$

Второй пример, демонстрирующий полезность подхода, — реализация интеллектуального управления трафиком в транспортной сети с пересечениями.

В работе [10] предложено использовать **моделирование по методу Монте-Карло**, причем управляющие решения вырабатываются в реальном времени путем выбора действия, которое производит наилучший статистический результат моделирования. Отмечается, что такой подход существенно лучше подхода на основе нейронных сетей.

Обзор ряда применений машинного обучения в задачах производственного управления представлен в работе [22]. Рассматривается **построение систем управления, основанных на правилах, отражающих опыт и поведение эксперта**, что связано с аппаратом символьного машинного обучения. Используется T-R (*Teleo-Reactive*) формализм Нильссона:

T-R последовательность есть агентная управляющая программа, направляющая агента к цели (Teleo) и учитывающая изменения окружающей среды (Reactive). В своей простейшей форме она состоит из упорядоченного множества продукционных правил.

Две T-R процедуры для управления мобильным роботом приведены на рис. 6.

```
goto(Loc)
     position = Loc                    →   nil
     heading = course(position, Loc)   →   move
     otherwise                         →   rotate

amble(Loc)
     position = Loc                    →   nil
     clear_path(position, Loc)         →   goto(Loc)
     otherwise                         →   amble(new_point(position, Loc))
```

Рис. 6. *Процедуры для управления мобильным роботом* [22]

Левая часть продукционного правила описывает проверку сенсорных данных, а правая — вызывает действующие процедуры. Первое правило процедуры `goto` постулирует, что при нахождении робота в нужном расположении ничего делать не нужно. Если курс робота такой, какой требуется, то двигаться вперед, иначе — повернуться. При повороте робота его курс всегда сравнивается с требуемым курсом.

Процедура `amble` описывает простой план для того, как избежать столкновения с препятствиями. Если путь чист (вторая продукция), то ничего не предпринимать; иначе определить новую точку размещения робота в стороне от препятствия и рекурсивно вызвать процедуру `amble`.

Обучение — извлечение правил из эмпирического опыта оператора — предполагает использования алгоритмов, основанных на построении деревьев решений, в частности $C4.5$ [18].

Известны многие другие работы, в которых описывается интеллектуальное управление расписанием призводственного процесса [17]. В для этой цели используется индуктивное моделирование (ID3 [19]), нейронные сети, обучаемые методом обратного распространения ошибки, нечеткая логика. В работе [20] использовалось 358 примеров для машинного обучения и 198 тестовых примеров. Ошибка обученных алгоритмов для рассматриваемых задач не превышала 10%.

Использование Q-обучения (*Q-learning*) оказалось полезным для интеллектуального адаптивного стохастического управления ресурсами [6]. *Q-learning* алгоритм Ваткинса оценивает функцию состояния-действия (*state-action*) Q следующим образом [9]:

$$Q_{t+1}(s,a) = (1-\alpha_t(s,a))Q_t(s,a) + \alpha_t(s,a)(R_M(s,a) + \gamma \max_{b \in U(s')} Q_t(s',b)),$$

где s' — состояние управляемого объекта, которое достигается из состояния s при выполнении управляющего действия a, произведенного в момент времени t; $U(s')$ — множество управляющих действий, которые можно произвести в состоянии s'; $R_M(s,a)$ — значение поощрения, получаемого за действие a в состоянии s; $\gamma \in (0,1)$ — параметр; $\alpha_t(s,a)$ — такая функция, что для любой допустимой пары (s,a) выполняются условия: $\sum_{t=1}^{\infty} \alpha_t(s,a) = \infty$ и $\sum_{t=1}^{\infty} \alpha_t^2(s,a) < \infty$.

Полученная в результате обучения функция $Q(s,a)$ позволяет определять оптимальную стратегию управления для каждого состояния s как

$$arg \max_a Q(s,a).$$

Q-обучение можно рассматривать как пример обучения, не использующего модель (*model-free learning*). В отличие от этой парадигмы, в интеллектульном управлении также используется **обучение, основанное на модели** (*model-based learning*) [5]. В этом случае агент использует внутреннюю модель, состоящую из двух частей. Первая — эволюционная модель — представляет собой причинно-следственную структуру "поведения" окружающей среды и обеспечивает прогнозирование последствий управляющих действий. Вторая часть — модель поощрения — обеспечивает оценивание текущего поощрения, связанного с конкретными ситуациями и действиями [5].

3. Логическое интеллектуальное управление

Обозначим управляемую систему S, а управляющую систему — U. Исходную информацию $I(S)$ об управляемой системе, необходимую для машинного обучения, будем считать заданной в виде таблицы T_{m_0r}, состоящей из m_0 строк — векторов $(x_1,\ldots,x_r)=\tilde{x}\in X^r$, описывающих m_0 наблюдений за состояниями системы S. Будем называть X^r — r-мерным пространством описаний (состояний) системы S. Будем также полагать, что таблица T_{m_0r} состоит из l подтаблиц

$$T^1_{m_1r}, T^2_{m_2r},\ldots,T^l_{m_lr}:\ T_{m_0r}=\bigcup_{1\le j\le l} T^j_{m_jr},$$

каждая из которых соответствует такому набору строк-наблюдений, когда система S находилась в одном и том же классе эквивалентных состояний X_j. Полагается, что заданы l классов экивалентности *разбиения множества* X^r: $X_1,X_2,\ldots,X_l:\ \bigcup\limits_{1\le j\le l} X_j=X^r$, и в таблице T_{m_0r} зафиксировано ровно m_j наблюдений из каждого класса $X_j,\ j=1,2,\ldots,l$.

Полагается также, что существует функция — показатель качества

$$\varphi: X^r\to V,$$

которая *apriori* неизвестна. Показатель качества принимает значения из множества V. Известными (заданными) являются значения показателя качества $v_1,\ldots,v_j,\ldots,v_l$ для состояний системы из классов $X_1,X_2,\ldots,X_l$, так что если $\tilde{x}\in X_j$, то $\varphi(\tilde{x})=v_j$.

Исходная информация $I(S)$ называется *правильной*, если для всех $\tilde{x}\in T_{m_0r}$ достоверно выполняется условие:

$$\tilde{x}\in T^j_{m_jr}\ \Leftrightarrow\ \varphi(\tilde{x})=v_j.$$

При заданном разбиении $X_1,\ldots,X_l$ "ссылочной" оценкой показателя качества $\varphi=\varphi(\tilde{x})$ можно считать значения функции $\psi: X^r\to\{1,2,\ldots,l\}$, которая частично задана таблицей T_{m_0r} с указанием в ней l подтаблиц, состоящих из наборов эквивалентных состояний вместе с информацией $v_1,\ldots,v_j,\ldots,v_l$ о значении показателя качества, взаимнооднозначно соответствующих натуральным числам (номерам) $1,2,\ldots,l$. Можно сказать, что функция ψ вычисляет номера значений показателя качества из множества $v_1,\ldots,v_j,\ldots,v_l$.

Пусть функции $P: X^r\to\{0,1\}^n$ и $f:\{0,1\}^n\to\{0,1,\ldots,l\}$ таковы, что

$$\psi=P\circ f. \tag{1}$$

Для существования композиции (1) необходимо и достаточно выполнения условия $n \geq [\log_2 l]$ [32]. Будем далее полагать, что это условие выполнено.

Обозначим $\tilde{y} = (y_1, \ldots, y_n)$ произвольный булевый вектор из множества $\{0,1\}^n = Y$. Тогда $P : X^r \to \{0,1\}^n$ является вектор функцией:

$$P(\tilde{x}) = (y_1(\tilde{x}), \ldots, y_n(\tilde{x})) = \tilde{y}(\tilde{x}).$$

Потребуем, чтобы выполнялось *условие биективности разбиений множеств* X^r *и* $Y = \{0,1\}^n$:

$$\forall j,k \in \{1,2,\ldots,l\} \quad Y_j \cap Y_k = 0; \quad \bigcup_{1\leq j\leq l} Y_j = Y;$$

$$\tilde{y}(\tilde{x}) \in Y_j \;\Leftrightarrow\; \tilde{x} \in X_j \;\Leftrightarrow\; \varphi(\tilde{x}) = v_j,$$

$$Y_j^{-1} = X_j.$$

Определение [32].

I^o Скалярные функции — предикаты $y_i : X^r \to \{0,1\}$ называются *признаковыми предикатами.*

II^o *Множество признаковых предикатов* $\{y_1, \ldots, y_n\}$ *называется допустимым*, если существует композиция (1).

III^o Логическим описанием класса j (ЛОК) называется формула D_j, построенная из предикатов $y_1, \ldots, y_n$ и связок $\wedge$, $\vee$, $\neg$ такая, что определяемая ею функция есть

$$F_{D_j}(\tilde{y}(\tilde{x})) = \begin{cases} 1, \; \tilde{x} \in Y_j; \\ 0, \; \tilde{x} \notin Y_j. \end{cases}$$

□

Таким образом вводятся функции F_{D_j}, $j = 1, \ldots, l$ — характеристические функции классов состояний, оцениваемых значениями показателя качества $v_1, \ldots, v_l$.

Если вычислить $P(\tilde{x})$ для всех $\tilde{x} \in T_{m_0 r}$, то таблица $T_{m_0 r}$ отобразится в булеву таблицу, состоящую из l подтаблиц:

$$B_{m_0 n} = \bigcup_{1\leq j\leq r} B^j_{m_j n},$$

где $B^j_{m_j n} \subset Y_j$.

Выбор признаковых предикатов является трудноформализуемой задачей и зависит от проблемной области. *Множество признаковых предикатов является допустимым, если в любой паре строк из разных подтаблиц эти строки различны* [32].

Когда признаковые предикаты выбраны, ЛОК D_j, $j = 1, 2, \ldots, l$, могут быть получены при помощи алгоритмов машинного обучения — решающих деревьев или

дизъюнктивных нормальных форм (ДНФ), синтезированных по частично заданной начальной информации — обучающей таблице B_{m_0n}.

Пусть ЛОК представлены в виде ДНФ. Рассмотрим подход к логическому интеллектуальному управлению объектом S, суть которого состоит в изменении управляющим алгоритмом U вектора $\tilde{y} = \tilde{y}(\tilde{x}) = (y_1(\tilde{x}), \ldots, y_n(\tilde{x}))$ с целью "перевода" системы S из текущего состояния в состояние с бóльшим значением показателя качества. Опуская для упрощения записи аргументы $\tilde{x}$, введем параметр t — шаг управления, и тогда $\tilde{y}(t)$ будет логическим описанием управляемой системы S на шаге t.

Пусть $\tilde{y}(\tilde{x})(t) = \tilde{y}_t \in Y_t$, и соответствующее состояние $\tilde{x}$ системы S имеет оценку показателя качества v_t.

Любое изменение хотя бы одной координаты булевого вектора $\tilde{y}_t$ будем называть *преобразованием* и обозначать

$$\tilde{y}_t \mapsto \tilde{y}_{t+1}.$$

Преобразование $\tilde{y}_t \mapsto \tilde{y}_{t+1}$ называется *целесообразным*, если $v_{t+1} > v_t$, и *оптимальным*, если обеспечивается максимальное увеличение показателя качества $v_{t+1} - v_t$ по всем возможным преобразованиям.

Управление осуществляется путем перевода системы из текущего состояния, которое принадлежит классу с некоторой достигнутой оценкой качества, в другой класс с более высокой или максимально возможно высокой оценкой качества. Основная идея логического управления — свести реализацию оптимального преобразования к сравнительному анализу конъюнкций, входящих в логические описания классов (ЛОК).

Пусть в результате машинного обучения по примерам таблицы B_{m_0n} получены ЛОК в ДНФ:

$$D_j(\tilde{y}) = \bigvee_{1 \le q \le s_j} K_q^j(\tilde{y}),$$

$$K_q^j(\tilde{y}) = y_{1,q}^{\sigma_{1,q}} \wedge y_{2,q}^{\sigma_{2,q}} \wedge \ldots \wedge y_{\tau_q,q}^{\sigma_{\tau_q,q}},$$

где

s_j — число конъюнкций в ДНФ $D_j(\tilde{y})$;

q — порядковый номер конъюнкции в ДНФ $D_j(\tilde{y})$;

τ_q — число логических сомножителей в конъюнкции $K_q(\tilde{y})$;

$z^\sigma = z$, если $\sigma = 1$, и $z^\sigma = \bar{z}$, если $\sigma = 1$.

Если $K_q^j(\tilde{y}_t(\tilde{x}_t)) = 1$, то показатель качества системы S в состоянии $\tilde{x}$ равен v_j, и все конъюнкции можно упорядочить по неубыванию значений показателя качества. Требуется преобразовать булевый набор (вектор) $\tilde{y}_t$, получая набор $\tilde{y}_{t+1}$ так, чтобы достигнуть наибольшего значения показателя качества. В результате такого

преобразования окажется, что $K_q^j(\tilde{y}_{t+1}(\tilde{x}_{t+1})) = 0$, $K_d^\mu(\tilde{y}_{t+1}(\tilde{x}_{t+1})) = 1$, и показатель качества примет значение $v_\mu > v_j$. Пусть затраты на такое преобразование набора $\tilde{y}_t \mapsto \tilde{y}_{t+1}$ равны $c_{qd}^{j\mu}(\tilde{y}_t)$.

Для текущего состояния $\tilde{y}_t$ требуется решить оптимизационную задачу

$$\max_{j,\mu}(v_\mu > v_j): \quad c_{qd}^{j\mu}(\tilde{y}_t) \ < \ C_t,$$

где C_t — текущее ограничение по затратам на шаге управления t.

Подходящие для этой цели оптимизационные модели представлены в работе [32].

4. Интеллектуальные агенты и управление

Под интеллектуальными агентами обычно понимают программы, которым человек делегирует свои функции с целью автоматического или полуавтоматического принятия решений. Агенты запоминают нужную для правильного функционирования информацию, обобщают ее, обучаются и могут самостоятельно генерировать управляющие воздействия или выдавать полезные рекомендации пользователям [30].

Интеллектуализация поиска информации в Интернете (*Web Browser Intelligent*), оптимизация обслуживания покупателей на основе сетевой информации, оптимизация функционирования больших высокопроизводительных многопользовательских вычислительных систем — вот некоторые примеры программ — интеллектуальных агентов.

Под автономностью интеллектуальных агентов понимают их способность контролировать самих себя, реализовывать самоуправление, реализуя целевое поведение: агенты имеют цель и действуют в соответствии с этой целью.

Агент — это объект, находящийся в некоторой среде, от которой он получает информацию, отражающую события, которые происходят в этой среде; агент интерпретирует события и исполняет команды, которые воздействуют на среду. Агент может содержать программные и аппаратные компоненты [33].

Под интеллектуальным агентом понимается программно или аппаратно реализованная система, которая обладает такими свойствами [33]:

— автономность – способность функционировать без вмешательства человека и осуществлять самоконтроль над своими действиями и внутренним состоянием;

— способность функционировать в сообществе с другими агентами, обмениваясь с ними сообщениями с помощью некоторого специального языка коммуникаций;

— реактивность – способность воспринимать состояние среды и своевременно реагировать на её изменения;

— способность генерировать цели и действовать, стремясь достигнуть их;

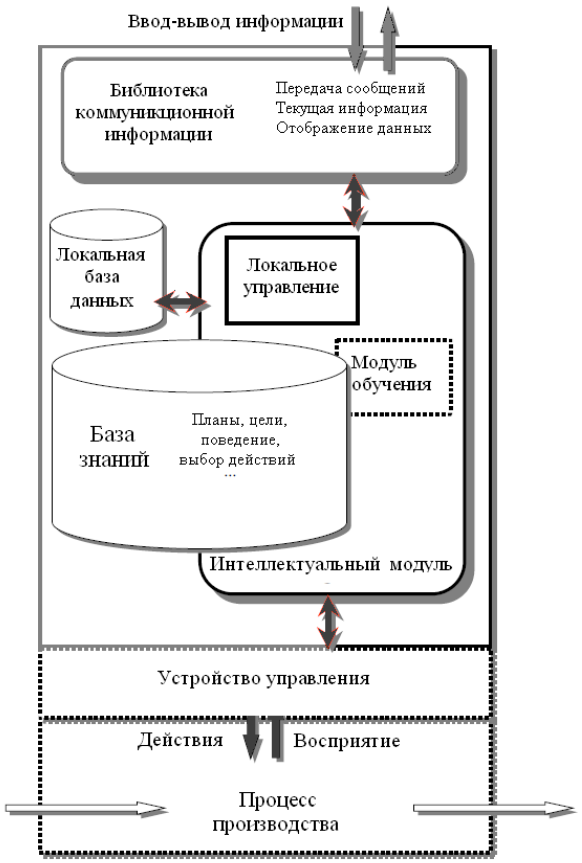

Рис. 7. *Архитектура производственного агента из работы Б. Кадара* [13]

— обладать знаниями о себе, среде и, возможно, других агентах; пополнять и модифицировать их.

Из приведенного определения интеллектуальных агентов видно, что управляющие функции для них являются основными, и причин рассматривать интеллектуальные агенты вне парадигмы интеллектуального управления нет.

Рассмотрим, например, архитектуру производственного агента из работы Б. Кадара [13], представленную на рис.7.

Эта архитектура на общем, концептуальном уровне соответствует представлению об интеллектуальном управленнии независимо от сферы приложений управляющих систем рассматриваемого класса. Но нужно заметить, что в каждом конкретном случае понадобятся многочисленные уточнения, главное из которых бутет определяться требуемой степенью автономности интеллектуальной управляющей системы.

Горизонтально-распростроняемые **интернетовские сервисы** (*Horizontally-scalable Internet services*) на компьютерных кластерах **представляют подходящую**

предметную область для применения интеллектуального управления. В частности, для целенаправленного управления, мониторинга, управления числом задействованных серверов. Для решения таких задач управления в работе [4] предлагается применять методы статистического управления, машинного обучения, имитационного моделирования.

Заключение

В результате обзора исследований в области интеллектуального управления можно сделать следующие выводы.

Интеллектуальное управление является самой интересной областью применения теоретической кибернетики в современных информационных системах. Здесь собираются воедино все наработанные приемы, методы и разделы теории: распознавание образов-состояний, формирование понятий, идентификация, оптимизация при неполной информации, теория самоорганизующихся автоматов. Сюда примыкают практически все методы исследования операций.

Круг практического применения методов интеллектуального управления чрезвычайно широк и включает: программы-агенты для сложных (особенно функционирующих в сетевой среде) информационных систем; агенты для производственного цехового управления; программы автоматизация принятия решений в региональных системах; программы управления в робототехнике; программы-исследователи для автоматических космических аппаратов и многое другое.

Наиболее сложными являются динамические интеллектуальные системы, способные перестраиваться по мере функционирования в среде, являющейся окружением объекта управления. Для реализуемости таких систем требуется моделирование механизма “забывания” информации, которая становится с течением времени неактуальной.

Важнейшими направлениями развития теории интеллектуального управления представляются: разработка специфических моделей памяти для агентов, алгоритмических баз знаний агентов, моделей динамической автоматической класификации состояний управляемых объектов и среды, автоматического изменения структур математических моделей и даже атоматической смены используемой агентом модели.

Список литературы

1. *Antsaklis P. J. Intelligent Control.* Written for the Encyclopedia of Electrical and Electronics Engineering. John Wiley and Sons, Inc. 1997.
   http://www3.nd.edu/ pantsakl/control.html
2. *Antsaklis P. J. Intelligent Learning Control*// IEEE Control Systems, 1995, Vol.15, No.3, pp. 5–7.

3. *Aydin M. E., Oztemel E. Dynamic job-shop scheduling using reinforcement learning agents*// Robot Auton Syst., 2000, Vol.33, No.2 – 3, pp. 169 –178.
4. *Bodik P., Griffith R. at al.Statistical Machine Learning Makes Automatic Control Practical for Internet Datacenters*. In Workshop on Hot Topics in Cloud Computing, 2009, 5 p. http://www.cs.berkeley.edu/ jordan/papers/bodik-etal-hotcloud09.pdf
5. *Botvinick M., Weinstein A.Model-based hierarchical reinforcement learning and human action control*// Model-based hierarchical reinforcement learning and human action control: Philosophical transactions of the Royal Society of London. Series B, Biological sciences, 2014,No.11, 9p. http://rstb.royalsocietypublishing.org/content/369/1655/20130480
6. *Csaji B. C., Monostori L. Adaptive Stochastic Resource Control: A Machine Learning Approach*// J. of Artificial Intelligence Research, 2008, Vol.32, No.2 – 3, pp. 453 –486.
7. *Doh H.-H., Yu J.-M. Kwon Y.-J. et al. Decision Tree Based Scheduling for Flexible Job Shops with Multiple Process Plans*// Int. J. of Mech., Aerospace, Industrial and Mechatronics Eng., 2014, Vol.8, No.3, pp. 615–621.
8. *Encyclopedia Britannica. Learning Control*. http://www.britannica.com/EBchecked/topic/334026/learning-control
9. *Even-Dar E., Mansour Y. Learning rates for Q-learning*// J. of Machine Learning Research, 2003, No.5, pp. 1–25.
10. *Galperin Gregory, Viola Paul. Machine Learning for Prediction and Control*. Rollout Project, Learning and Vision Group Artificial Intelligence. Laboratory M.I.T., 1998, 4pp.
11. *Gers F. A., Schmidhuber J., Cummins F. Learning to Forget: Continual Prediction with LSTM*// Neural Computation, 2000, Vol. 12, No.10, pp. 2451–2471.
12. *Hamdi-Cherif A. Knowledge Base Learning Control System - Part 1: generic architecture, Part 2: Intelligent Controller*. AIKED'12 Proceedings of the 11th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases, 2012, pp. 67–76.
13. *Kadar Botond. Intelligent approach to manage changes and disturbances in manufactoring systems*. Ph.D. Thesis. Computer and Automation Research Institute, Hungarian Academy of Sciences. Budapest, 2001, 13p.
14. *Kim C.-O., Min H.-S., Yih Y. Integration of inductive learning and neural networks for multi-objective FMS scheduling* // Int. J. Prod. Res., 1998, Vol. 36, No. 9, pp. 2609 – 2626.
15. *Konaka E. Design of Discrete Predictive Controller Using Approximate Nearest Neighbor Method*. Preprints of the 18th IFAC World Congress Milano (Italy) August 28 - September 2, 2011, 6 p.
16. *Piramuthu S., Raman N., Shaw M.J. Learning-based scheduling in a flexible manufacturing flow line*//IEEE Trans. Eng. Manag., 1994, Vol.41, No. 2, pp. 171 – 182.
17. *Priope P., De La Fuente D., Gomes A., Puente J. A review of machine learning in dynamic scheduling of flexible manufacturing systems* // Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 2001, Vol.12, Issue 3, pp. 251-263.
18. *Quinlan J. R. Improved use of continuous attributes in c4.5* // J. of Artificial Intelligence Research, 1996, No.4, pp.77–90.
19. *Quinlan J. R. Induction of Decision Trees* // Machine Learning, 1986, No.1, pp.81–106.

20. *Quiroda L. A., Rabelo L. C. Learning from examples: a review of machine learning, neural networks and fuzzy logic paradigms* // Computers & Industrial Engineering, 1995, Vol.29, Issues 1–4, pp.561–565.
21. *Sadigh Dorsa, Kim Eric S., Coogan Samuel, Sastry S. Shankar, Seshia Sanjit A. A Learning Based Approach to Control Synthesis of Markov Decision Processes for Linear Temporal Logic Specifications*// arXiv:1409.5486, 2014, 6 p.
http://arxiv.org/pdf/1409.5486v1.pdf
22. *Sammut Claude. Automatic construction of reactive control systems using symbolic machine learning*// Knowledge Engineering Review, 1996, Vol.11, pp. 27 – 42.
23. *Schaffernicht E., Stephan V. et al. Machine Learning Techniques for Selforganizing Combustion Control*. In: Proc. 32nd Ann. Conf. on Artificial Intelligence (KI 2009), Paderborn, 2009, pp. 395-402.
24. *Shnits Boris, Sinreich David Controlling flexible manufacturing systems based on a dynamic selection of the appropriate operational criteria and scheduling policy*// Int. J. Prod. Res., 2004, Vol.42, Issue 17, pp. 3457 – 3472.
25. *Shaw M.J., Park S., Raman N. Intelligent scheduling with machine learning capabilities: the induction of scheduling knowledge*//IIE Transactions, 1992, Vol.24, Issue 2, pp. 156 – 168.
26. *Shiraz G. M., Sammut C. Combining Knowledge Acquisition and Machine Learning to Control Dynamic Systems*. 15th Int. Joint Conf. on AI, Nagoya Japan, Morgan Kaufmann, 1997, pp. 908–913.
27. *Sommer N.,Ralescu A. Towards a Machine Learning Based Control of Musical Synthesizers in Real-Time Live Performance*. Proc. of the 25th Modern Artificial Intelligence and Cognitive Science Conf., Spokane, Washington, USA, April 26, 2014, pp. 61–67.
28. *Su C.-T.,Shiue Y.-R. Intelligent scheduling controller for shop floor control systems: a hybrid genetic algorithm/decision tree learning approach*// Int. J. Prod. Res., 2003,Vol. 41, No. 12, pp. 2619–2614.
29. *Sutton R. Artificial intelligence as a control problem: Comments on the relationship between machine learning and intelligent control*. IEEE International Symposium on Intelligent Control, 1988, pp. 500–507.
30. *Tran Hanh, Tran Thaovy Intelligent Agent.*
http://groups.engin.umd.umich.edu/CIS/course.des/cis479/ projects/agent/Intelligent agent.html
31. *Xu Jianming, Mingxuan Sun, Li Yu LMI-Based Synthesis of Robust Iterative Learning Controller with Current Feedback for Linear Uncertain Systems*// International Journal of Control, Automation, and Systems, 2008, Vol. 6, No. 2, pp. 171–179.
32. *Донской В. И. Логическое управление плохо формальзованными системами*// Динамические системы, 1985, вып.4, с. 90 – 96.
Donskoi V. I. Logical control of poorly formalized systems // J. of Soviet Mathematics, 1992, Vol. 60, No. 2, p. 1402 – 1406.
33. *Слабое и сильное определение интеллектуального агента.* Портал искусственного интеллекта — www.AIportal.ru.
Strong and weak definition of intelligent agent . AI portal – www. AIportal.ru.
http://www.aiportal.ru/articles/multiagent-systems/weak-and-strong-intelligent-agent.html

Статья поступила в редакцию 02.12.2014