Veritone CDI Architeture

Wolf Kohn

Veritone CDI Architecture

1 Introduction to CDI

The *Cooperative Distributed Inferencing (CDI)* system is a unique advanced technology enabling near-optimal and near real-time decision making on large-scale, heterogeneous and distributed information with the use of rules. CDI integrates absolute, hard and soft rules within complex knowledge-based decision support systems to achieve performance goals while satisfying various requirements from natural or governing laws, policies, and best practices.

The CDI system has a *Distributed Architecture (DA)*, consisting of a network of *Decision Elements (DEs)* that work together to resolve queries and identify Pareto efficient states. The decision elements access shared information from both an *Internal Heterogeneous Database (IHDB)* and *External Knowledge Base (EKB*). Each decision element solves a query using optimal control theory, starting with a technique called analytic continualization - transforming the query and rules into differential equations whose dependent variables represent internal variables and parameters of the rules. The decision elements in the architecture are synchronized via a *Pareto multi-criteria* optimization strategy.

CDI features a self-adapting and learning design. Since CDI converts the original query into an optimal control problem, it can use feedback from the environment (e.g., external sensors or internal knowledge updates from other DEs) to refine its internal model; the Hamilton-Jacobi-Bellman equation will be updated to reflect new information and automatically form soft rule-like constraints internally.

CDI is particularly applicable when the system has large-scale heterogeneous data, rules from government compliances and/or business requirements, and the need to make near real-time decisions. Healthcare and energy are two such applications.

2 Contents, Notation and Abbreviations

2.1 Table of Contents

2.2 Notation

The following table is intended to summarize the notation that is used throughout the document. This is a work-in-progress.

Current notation	Recommended notation	First occurrence	Comments
t	same	6.2.2	Notation for algorithmic time.
$q(t)$	same	15	Notation of canonical coordinate vector for entire system.
q	same		Notation of canonical coordinate vector dropping time dimension.
$q^{(f)}$	same		Notation of canonical coordinate vector for specific function f.
$\dot{q}$	same		Notation of first time derivative of canonical coordinate vector, $dq(t)/dt$
$\ddot{q}$	same		Notation of second time derivative of canonical coordinate vector, $d^2q(t)/dt^2$
h	same		Notation of HEAD of Horn clause.
$\varphi(q)$	same		Notation of generic proposition.
$\sigma(q)$	same		Notation of generic proposition alternate to φ.
T_i	same		Notation of the TV of a soft rule.

$\check{r}(q;\varphi,\sigma)$	same		Generic equational form relating two propositions.
$\check{\varphi}(q)$	same		Notation of the equational form of $\varphi(q)$.
$\varphi_Q(q)$	same		Notation for proposition defined by the query.
$\check{\varphi}_Q(q)$	same		Notation for equation defined by the query.
$J(q)$	same		Notation for minimization function for the query.
$\mathcal{L}$	same		Notation for static Lagrangian
$\mathcal{L}_k^{(o,T)}$	same		Notation for total static Lagrangian for DE_k.
q	same		
$\{p_a\}$	same		
$u^{(k)}$	same		
$H_k^{(o)}$	same		Primary Hamiltonian for the absolute rules for DE_k.
$H_k^{(A)}$	same		Hamiltonian for the Tellegen agent of the total Hamiltonian's rules.
$H_k^{(T)}$	same		Total Hamiltonian for DE_k.

2.3 Abbreviations

Current abbreviation		First occurrence
CDI	Cooperative Distributed Inferencing	Section 1
DE	Decision Element	Section 1
IHDB	Internal Heterogeneous Database	Section 1
EKB	External Knowledge Base	Section 1
DA	Distributed Architecture	Section 1
REI	Rule Entry Interface	Section 6
RE	Rule Editor	Section 6
SII	Sensor Ingestion Interface	Section 6
RCE	Rule Conversion Engine	Section 6
QLI	Query Language Interface	Section 6
MFG	Minimization Function Generator	Section 6
QRE	Query Response Engine	Section 6
PMOE	Pareto Multi-criteria Optimization Engine	Section 6
OP	Optimization Process	Section 6.1
LER	List of External Repositories	Section 6.1
KC	Knowledge Component	Section 6.2.1
TV	Truth Valuation	Section 6.2.2
API		Section 6.3
IE	Inference Engine	Section 6.8
PSE	Programmable Search Engine	Section 6.8
IRB	Inference Rule Base	Section 6.8
UI	User Interface	Section 6.8

NI	Network Interface	Section 6.8
VB	Variable Buffer	Section 6.8.2
ARB	Active Rule Buffer	Section 6.8.2
IP	Inference Process	Section 6.8.2
IR	Inference Rules	Section 6.8.5

3 Overview of the Document

This document introduces and specifies the architecture for the *Cooperative Distributed Inferencing* (CDI) system. The primary instance of this is the *Distributed Architecture (DA)* for resolving *queries* by accessing both an *Internal Heterogeneous Database (IHDB)* populated by a special class of Horn Clause rules and external data sources referred to as *sensors*.

The architecture *implements* a network of active devices at its nodes. Active devices may be passive, generative, or both. These devices are called *Decision Elements (DEs).* The DEs *cooperate* in the resolution of a query posed to one or several of them. The DEs in a given DA are referred to as the *team.*

Every DE in a team is programmed to *transform* rules in its domain, determined by a posed query, into an *ordinary differential equation*, whose dependent variables represent internal variables and parameters. The dependent variables include unknowns of the query posed to the DE. The DEs in the architecture are synchronized via a *Pareto multi-criteria* optimization strategy.

This document reviews the components of the CDI system including:

- Application requirements that the system is designed to accommodate.
- Functional requirements that satisfy the application requirements and pertain directly to the construction and operation of the system components.
- Subcomponents, which are necessary to implement the functional requirements.
- Limitations that highlight noteworthy constraints that are inherent to the specified implementation of the architecture.
- Architectural flow describing key aspects of the architecture that indicate how the system is to be constructed given the specified essence and key behavior of the subcomponents.

- Software realization of the architecture that describes the key pieces of software necessary for system implementation.
- Data that describes the kinds of data the system is expected to accept as input and produce as output.
- Data exchange protocols reference key data types and structures that need to be exchanged across the system and the protocols for exchange.
- Environment describing the particulars of the environments that the system will be able to operate in and therefore should be tested in.
- Testing that describes how the system should be tested given the data and operating environments.

4 Application Requirements

The application requirements discussed in this architecture document articulate the salient aspects of the architectural strategy, approach and design. Key reference documents are listed.

The key areas of application requirements are:

- System specification that describes *what the system should be able to do*.
  See Section 4.1.
- System operation that describes in *what contexts the system should be able to operate*. See Section 4.2.
- System performance objectives that describes *how well the system should perform in the various contexts*. See Section 4.3.

4.1 Application requirements pertaining to *system specification*

4.1.1 Ability to integrate data that may include millions of variables and hundreds of thousands of constraints

4.1.2 Ability to provide data integration over distributed network which assimilates and integrates information over time across the network as needed

4.1.3 Ability to specify queries over a broad range of languages

4.1.4 Ability to specify queries of a broad range of complexity

4.1.5 Ability to provide best known response to queries at the local level

4.2 Application requirements pertaining to *system operation*

4.2.1 Ability to operate in a variety of environments including EC2, Azure, and local deployments

4.3 Application requirements pertaining to *system performance objectives*

4.3.1 Ability to provide responses to queries at intervals as small as one millisecond

5 Functional Requirements

Functional requirements match the key application requirements and describe specifically what the software should achieve.

6 Subcomponents

Subcomponents are fundamental parts of the architecture that perform particular roles. This section contains descriptions of each of the subcomponents of the architecture. The subcomponents are:

1. The Distributed Architecture (DA).
2. The Internal Heterogeneous Database (IHDB).
3. The Rule Entry Interface (REI).
4. The Rule Editor (RE).
5. The External Knowledge Base (EKB).
6. The Sensor Ingestion Interface (SII).
7. The Rule Conversion Engine (RCE).
8. The Decision Element (DE).

9. The Query Language Interface (QLI).
10. The Minimization Function Generator (MFG).
11. The Query Response Engine (QRE).
12. The Pareto Multi-Criteria Optimization Engine (PMOE).

6.1 Distributed Architecture (DA)

The DA, illustrated in Figure 6.1-1, is a network of computing devices as its nodes called DEs that interact and collaborate in the resolution of a query posed by one of them. The DEs access data and information stored locally in the Internal Heterogeneous Database (IHDB). The DEs also communicate over the network with sensors and an External Knowledge Base (EKB) for real-time data and rules. The DEs implement a distributed, dynamic optimization process, herein referred to as *the optimization process* (OP). The OP implements an optimization process that computes an answer to the active queries *as a function* of data stored in both the IHDBs and EKBs. These repositories of the data are needed to implement the OP given a query.

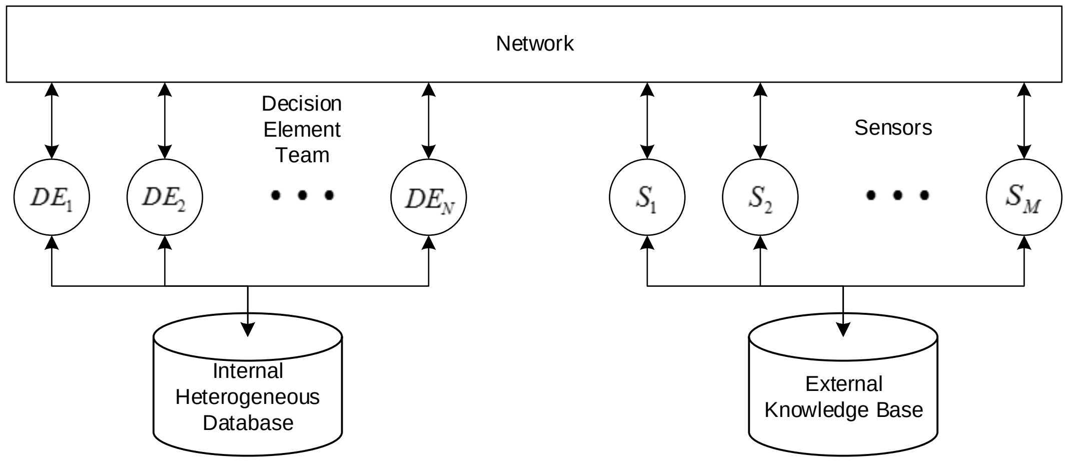


Fig. 6.1-1. Distributed Architecture.

The DA's block diagram is shown in Fig. 6.1-1. The rest of the document is devoted to describe the functional characteristics of this architecture and in particular, the DEs, IHDB, and the sensors. In particular the document will address the following concepts:

1. The DA
2. A process for resolving queries by accessing the IHDB and External Knowledge Bases (EKBs) through sensors
3. The constitution of DEs
4. A query and corresponding rules transformation into an ordinary differential equation

5. The orchestration of a team of DEs through a Pareto multi-criteria optimization strategy

Figure 6.1-2 illustrates how the DEs communicate over the network to get information from a variety of knowledge sources (including sensors, information in a social network, dictionary, Wikipedia, etc.), and also access their IHDBs to respond to a query.

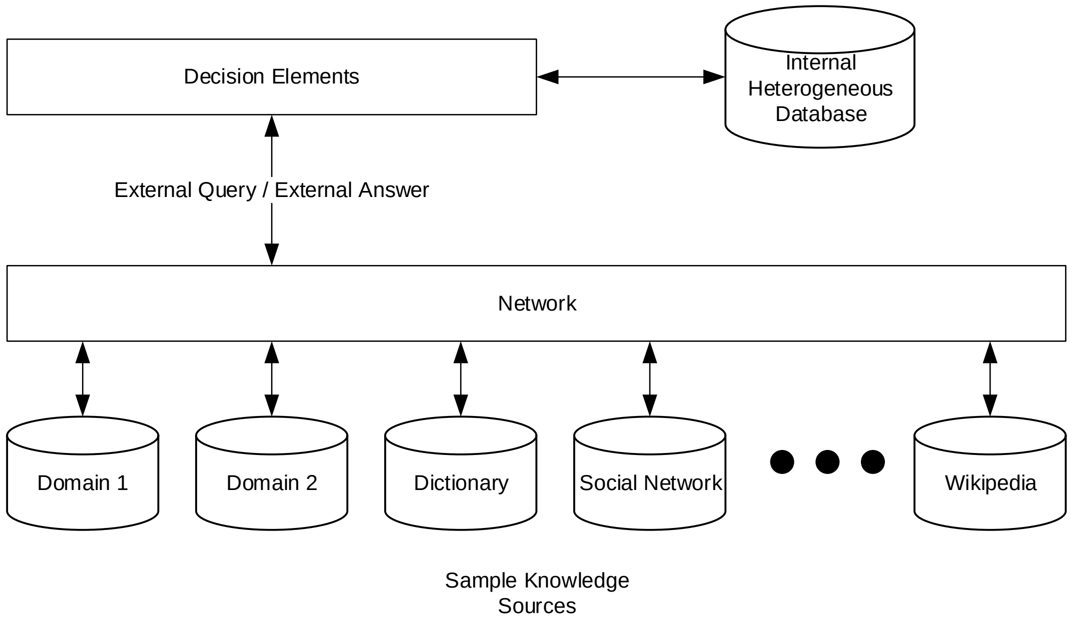


Fig. 6.1-2. Knowledge Bases.

A DE has a *list of external repositories* (LER). Each entry in an LER includes 1) a *protocol*, 2) a *heading sub-list*, and 3) a *translation grammar*. Each protocol entry prescribes the access procedure to the corresponding knowledge repository. Each heading sub-list entry contains a summary of the knowledge contents of the corresponding repository. Finally, each translation grammar entry provides a procedure for converting knowledge elements of the corresponding repository into the rule representation in the IHDB of the DE. This representation is discussed below.

The EKBs, illustrated in Figure 6.1-2, are a collection of public or private repositories of knowledge relevant to the DE posing a query.

6.2 The Internal Heterogeneous Database (IHDB)

6.2.1 Composition of IHDB as a set of knowledge components (KCs)

The IHDB encodes knowledge and data provided by the DEs regarding the implemented application. The IHDB is divided into *knowledge components (KCs)*. Each KC is consulted and updated by a DE in the DA. Any pair of KCs may have an overlapping set of rules by which they operate, but there is no *a pr*iori constraint on intersections or inclusion. The collection of KCs constitutes the existing knowledge of the system, and contributes to the IHDB, as illustrated in Figure 6.2-1.

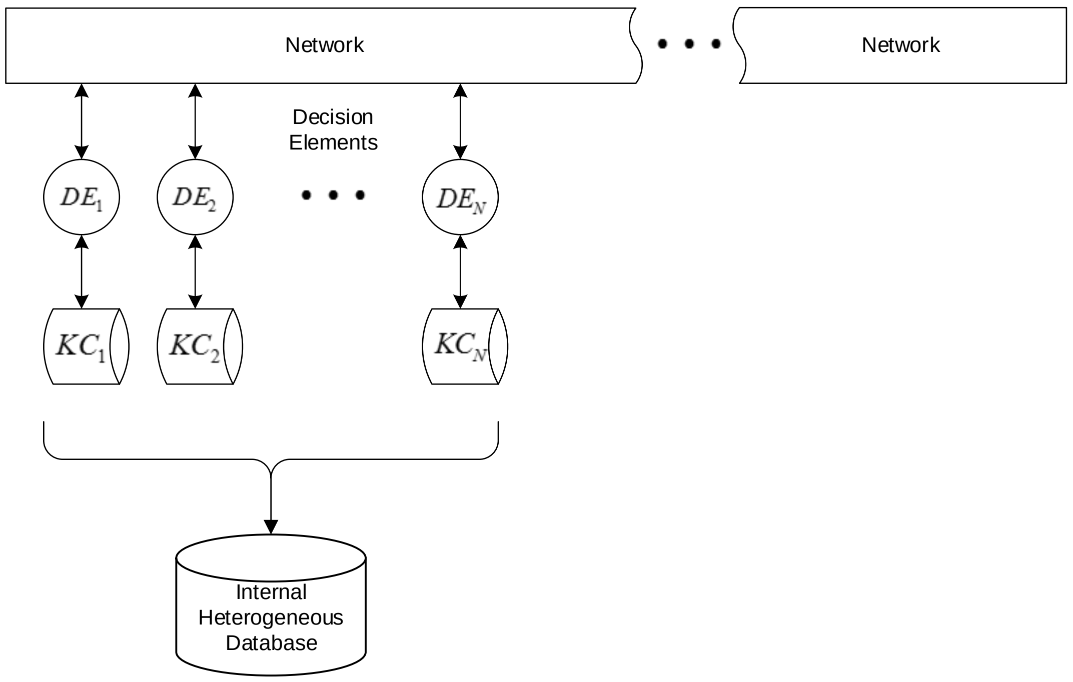


Fig. 6.2-1. Representation of Knowledge Components for Decision Elements across the Distributed Architecture.

6.2.2 Algorithmic formulation of a rule

A KC is a collection of *rules,* written in a *restrictive* Horn clause format. Horn clauses are consistent with a Prolog-based representation. The format of a Horn clause is given in the form of IF-THEN representation,

$$p_1 \wedge p_2 \wedge \cdots \wedge p_K \cdots \rightarrow p$$

where $p_i, i = 1, \ldots, K$ and p are propositional variables that are either True or False (0 or 1). The rules are logic entities, that, when *instantiated*[1], obtain a logic value. Typically, the logic values a rule can obtain are binary, however, more generally, the logic values are taken from the interval [0,1].

The entire system of rules is evaluated using *variables* and *parameters* that are collectively referred to as the *generalized coordinates (state) of the system* and are indexed as follows

$$q(t) = \{q^{(1)}(t), \ldots, q^{(N)}(t)\}. \tag{6.2-1}$$

The time argument t refers to the algorithmic time of the system, which means that it is a continuous index with respect to the evolution of the system. There is no requirement that it correspond to a physical aspect of the system, although this may naturally occur. Physical time may be represented specifically by a canonical coordinate of choice $q^{(i)}(t)$. Alternatively, we may refer to the q's without expressly stating the independent time argument and write

$$q(t) = \{q^{(1)}, \dots, q^{(N)}\}.$$

(6.2-2)

Then we should also note that the time derivatives are denoted as

$$\dot{q} = \frac{dq(t)}{dt}, \qquad \ddot{q} = \frac{d^2q(t)}{dt^2}$$

(6.2-3)

These coordinates are referred to variously as q, depending on the context and the expected arguments of the function to which they are applied. When it is necessary to distinguish between more than one q in equational form we generally write q_f where f denotes the reference function or appropriate domain. *Typically, we assume without loss of generality the entire set of canonical coordinates q is an argument to any function, term or proposition.* In practice, we may further assume it is possible to apply the particular required coordinates as need to mathematical construct in question.

The rules in each knowledge component are of three types*: absolute rules*, *hard rules*, and *soft rules*. Absolute rules and hard rules take logic value 0 (false) or 1 (true) when instantiated. Soft rules take any value in the interval [0,1].

The format of the restrictive Horn Clauses in the IHDB is illustrated in Fig. 6.2-2. A Horn Clause is an object composed of two objects a *HEAD* and a *BODY* connected by backward implication ($\Longleftarrow$). The logic implication transfers the logic value of the *BODY* to the *HEAD*. If the rule is an absolute rule or a hard rule, the logic value is 1(if the *BODY* is logically true) or 0 (if the *BODY* is logically false). If the rule is a soft rule, the logic value transferred by the body is any number in $[0, 1]$.

The *HEAD* is a data structure composed of two objects: A *name*, *h*, and a *list of arguments* described by the argument vector $q = \left(q^{(1)}, \dots, q^{(N)}\right)$. The list of arguments includes *variables* and *parameters.* The variables take values in the *domain* of the rule and the parameters are constants passed to the rule and unchanged by the instantiation of the rule. The domain of the rule is a set of values that each of its variables can take. In general, variables can take values over numerical or symbolic domains. As an example, a symbolic domain can be a list of diseases. A numeric domain can be a set of pairs of numbers representing blood pressure.

For the applications of CDI, the domains for variables are: real numbers, complex numbers (floating point and floating point complex numbers), integer numbers, binary numbers and symbolic token on finite domains.

The *BODY* of a clause is a data structure composed of one or more *terms,* denoted $\varphi_i(q)$. The composition operation is *extended-and*, denoted by: $\wedge$. The extended-and works as a regular *and* in absolute rules and hard rules and as a *functional product*[2] on soft rules.

A rule with a head but not a body is called a *fact.* A fact's truth value is determined on the basis of the instantiation of its variables.

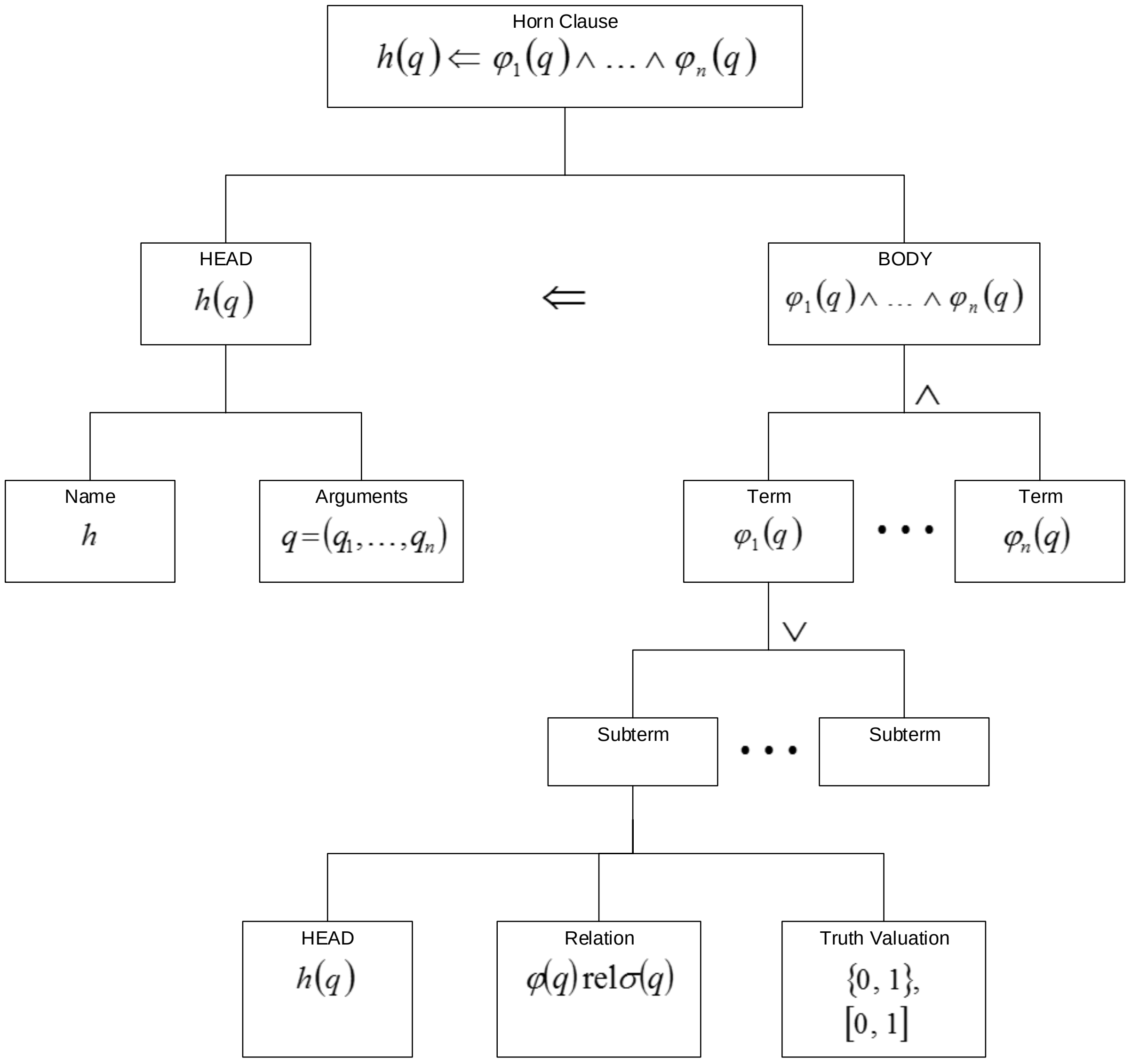


Fig. 6.2-2. Horn Clause (rule).

Each term in the body of a rule is an extended disjunction (*or* denoted by ∨) of *sub-terms*. The ∨ operator behaves like the *standard-or* for absolute and hard rules and behaves in a functional form, described later, when connecting sub-terms encoding heads of soft rules.

A sub-term is either the *HEAD* of a rule, a *relation* or a *truth valuation* (TV). When it is a *HEAD* it may have the same name as the one in the *HEAD* of the rule but with different arguments. This provides a recursive mechanism for rule evaluation.

When a rule has a sub-term that is the head of another rule it is said that the two rules are *chained* together by the corresponding sub-term. Note that a rule can be chained to several rules via corresponding sub-terms.

6.2.3 Constraint domains

Constraint domains augment the *BODY* clause of Horn clauses to facilitate dynamic programming. Constraints are specified as a relationship between terms. Define the relationship between two terms

$$\varphi(q) \text{ rel } \sigma(q). \tag{6.2-4}$$

as a member of the following set

$$\text{rel} \in \{=, \neq, \leq, \geq \text{ , statistical propagation, symbolic}\}. \tag{6.2-5}$$

A relation can be of two types *numeric* or *symbolic.* Numeric relations establish equational forms between two functional forms. (For the initial phase only polynomial and affine linear functional forms will be considered.)

In general, an equational form is a set of one or more *relations*. For numeric relations, $\varphi(q) \text{ rel } \sigma(q)$, $\text{rel} \in \{=, \neq, \leq, \geq, <, > \text{ , statistical propagation}\}$. Table 1 gives the relations considered and their symbols.

Numeric Relation	**Symbol**	**Code Form**
Equality	$=$	$\varphi = \sigma$
Disequation	$\neq$	$\varphi\backslash = \sigma$
Less-inequality	$<$	$\varphi < \sigma$
Less-Equal	$\leq$	$\varphi =< \sigma$
Great-inequality	$>$	$\varphi > \sigma$
Great-Equal	$\geq$	$\varphi >= \sigma$

Table 6.2-1. Numeric Relations.

The adopted code forms are the ones used in *constraint logic programming*.

A symbolic relation can be of two types: *inclusion* and *constraint.* Inclusion relations are of the form:

$$x \in Set \tag{6.2-6}$$

where x is a variable or a parameter, $\in$ is the inclusion symbol and Set is a set of symbolic forms or a set of numbers or a composite set of the form shown in Table 6.2-2.

Composite Set	**Symbol**	**Code Form**
Intersection	$\cap$	$Set1/\backslash Set2$
Union	$\cup$	$Set1\backslash/Set2$
Complement	$\backslash$	$\backslash Set$

Table 6.2-2. Composite Sets

Constraint forms of the symbolic relational type may be one or a set of the forms presented in Table 6.2-3. For numeric relations, $\varphi(q)$ rel $\sigma(q)$, rel $\in \{=, \neq, \subset, \supset, \subseteq, \supseteq\}$.

Symbolic Relation	**Symbol**	**Code Form**
Equal	$=$	$\varphi\# = \sigma$
Not Equal	$\neq$	$\varphi\#\backslash = \sigma$
Is Contained	$\subset$	$\varphi\# < \sigma$

Contains	$\supset$	$\varphi\# > \sigma$
Is Contained or Equal	$\subseteq$	$\varphi\# =< \sigma$
Contains or Equal	$\supseteq$	$\varphi\# >= \sigma$

Table 6.2-3. Constraint forms

A TV is either a variable or a constant with values in the interval $[0, 1]$. The TV of a Horn Clause that is an absolute rule or a hard rule can only take two values: 1 or 0. The TV when instantiated is 0 or 1. If the TV for an absolute or hard rule is 1, the rule is said to be in *inactive state;* if the TV is 0, the rule is said to be in *active state.*

The TV, T_i, of a soft rule satisfies

$$0 \le T_i \le 1. \tag{6.2-7}$$

If T_i above satisfies,

$$T_i \ge T_{threshold} \tag{6.2-8}$$

the soft clause is said to be in *inactive state*. If

$$T_i < T_{threshold}, \tag{6.2-9}$$

the soft clause is said to be in *active state,* where $T_{threshold}$ is a constant in [0,1] defined for each soft clause. The default value is 0.5.

This concludes the description of the knowledge representation. The *instantiation process* of the goal in a DE, as function of its knowledge base, is carried out by the inference engine of the DE (see Fig. 6.8-2). This process is the central component of CDI and is described later on the document.

6.2.4 Summary of terminology

The following table summarizes the terminology we have just reviewed.

Reference term	**Definition**
Proposition	Defined as a construct as in the propositional calculus where the

	proposition takes on the value of true or false.
Term	Recursively according to its assigned sub-term.
sub-term	A sub-term may be a Horn clause, a relation between two other sub-terms or an extended truth valuation depending on the context of absolute, hard or soft rules. In the case of absolute and hard rules it may be evaluated as a proposition. In the case of soft rules it takes a value on the interval [0,1] and is considered to be active or true in the case that it exceeds its specific threshold.
Horn clause	A disjunction of terms with *at most* one positive term.
definite clause	A Horn clause with exactly one positive term.
goal clause	A Horn clause with no positive terms.
Fact	A definite clause with no negative terms.
Head	The positive term of a definite clause.
inactive state	The case when a rule will *not* apply for constrained optimization.
active state	The case when a rule will apply for constrained optimization.
truth value, TV	The value that is used to determine whether a rule is active or inactive.

Table 6.2-4

6.2.5 Horn clause example

The following example illustrates a Horn clause:

$$\begin{aligned}\text{has\_fever}(name, temperature, white\_count, heartrate, blood\_pressure) \\ \Leftarrow (temperature > 37) \\ \land \big((heartrate \geq 70) \lor bp(name, temperature, blood\_pressure) \\ \lor wc(name, white\_count)\big)\end{aligned}$$

(6.2-10)

The clause establishes under which conditions the patient of name *name,* has a fever. The name of the rule is "has_fever", and the arguments are: $name$, $temperature$, $white\_count$, $heartrate$, $and\ blood\_pressure$. Of these arguments, $name$ is a parameter and the other arguments $(temperature, white\_count, heartrate, blood\_pressure)$ are variables in the clause. The Body has two terms, $\varphi_1(q)$ and $\varphi_2(q)$, and the second term has three sub-terms.

When the arguments are instantiated, they represent, respectively, the name of the patient, current body temperature, white blood cell count, heart rate range, and blood pressure.

The clause body includes other clauses: bp (blood pressure) and wc (white count).

This completes the specification of the rule-based framework. The next step is to specify a complete process for converting all rules of this form to a set of equations.

6.3 Rule Entry Interface (REI)

The Rule Entry Interface provides a mechanism for:

1. Providing an API for the entry of rules into the IHDB.
2. Validating the specification of rules to be inserted into the IHDB.
3. Routing the rules to the appropriate DEs for insertion to their respective KCs.

6.4 Rule Editor (RE)

The Rule Editor allows users to specify rules associated with the systems to be interrogated.

6.5 External Knowledge Base (EKB)

Need to discuss the following:

- Is it distributed?
- What is persisted?
- Where is it persisted?
- What is the relation to the IHDB? E.g. are they architecturally co-located?

- What is a sensor? This is a broadly defined notion, but ultimately we will need to specify some key instantiations.
- What does the bus look like?
- Should we have sensors collocated with DE's in some cases?

6.6 Sensor Ingestion Interface (SII)

TBD.

6.7 Rule Conversion Engine (RCE)

The rule conversion engine converts rules of the IHDB into equations.

6.7.1 Method for specification of a simple term as an equation

Consider the term $\varphi(q)$ with the following truth assignment,

$$\varphi(q) = \begin{cases} T & q \in \mathcal{D}_\varphi \\ F & q \notin \mathcal{D}_\varphi \end{cases}$$

(6.7-1)

Then we can define the set of arguments that yield positive truth assignment,

$$\{q \in \mathcal{D}_\varphi | \varphi(q) \longleftarrow T\}.$$

(6.7-2)

and define the corresponding equation $\breve{\varphi}(q)$ of the term $\varphi(q)$ as

$$\breve{\varphi}(q) = \begin{cases} 1 & \varphi(q) \longleftarrow T \\ 0 & \varphi(q) \longleftarrow F \end{cases}$$

(6.7-3)

and then extend the range of $\breve{\varphi}(q)$ to the closed unit interval

$$\breve{\varphi}(q) \rightarrow [0,1].$$

(6.7-4)

Revisiting the taxonomy of absolute, hard and soft rules, we recognize that soft rules (terms in this example) can take values along the interval

$$0 \leq \breve{\varphi}(q) \leq 1$$

(6.7-5)

whereas, absolute and hard rules should satisfy the additional constraint $\breve{\varphi}(q) \rightarrow \{0,1\}$

$$\breve{\varphi}(q)\big(1 - \breve{\varphi}(q)\big) = 0.$$

(6.7-6)

6.7.2 Conversion of the fundamental clauses of propositional calculus to equations

Define the following notation for the propositional calculus.

Symbol	**Function**
$\wedge$	And
$\vee$	Or
$\Longrightarrow$	Implication
$\sim$	Not
$\exists$	Exists
$\forall$	All
Statistical propagation	

Table 6.7-1

Theorem 6.7.1. Given the method for the specification of equations from propositions, we prove the following transformations.

Proposition	**Equation**
$\sim\varphi(q)$	$1 - \check{\varphi}(q)$
$\varphi(q) \wedge \sigma(q)$	$\check{\varphi}(q) \cdot \check{\sigma}(q)$
$\varphi(q) \vee \sigma(q)$	$\check{\varphi}(q) + \check{\sigma}(q) - \check{\varphi}(q) \cdot \check{\sigma}(q)$
$\varphi(q) \Longrightarrow \sigma(q)$	$1 - \check{\varphi}(q) + \check{\varphi}(q) \cdot \check{\sigma}(q)$
$\varphi_1(q) \wedge \varphi_2(q) \wedge \cdots \wedge \varphi_{k-1}(q) \wedge \varphi(q) \Longrightarrow \varphi(q)$ (tail recursive)	$\check{\tilde{\varphi}}(n,q) = \dfrac{\check{h}(n-1,q)}{\check{\tilde{\sigma}}(n,q)\check{\tilde{\varphi}}(n-1,q) - 1}$

Table 6.7-2

6.7.2.1 *Proof by enumeration for equational representation of negation*

Define the function $\check{r}(q;\varphi,\sigma)$ which represents the equation corresponding to negation ($\sim$). Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \rightarrow 1$ and $F \rightarrow 0$.

$\varphi(q)$	$\sim\varphi(q)$	$\check{r}(q;\varphi,\sigma) = 1 - \check{\varphi}(q)$
T	F	$0 = 1 - 1$
F	T	$1 = 1 - 0$

Table 6.7-3

6.7.2.2 *Proof by enumeration for equational representation of conjunction*

Define the function $\check{r}(q;\varphi,\sigma)$ which represents the equation corresponding to conjunction $(\wedge)$. Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \rightarrow 1$ and $F \rightarrow 0$.

$\varphi(q)$	$\wedge$	$\sigma(q)$	$\check{r}(q;\varphi,\sigma) = \check{\varphi}(q)\cdot\check{\sigma}(q)$
T	T	T	$1 = 1\cdot 1$
T	F	F	$0 = 1\cdot 0$
F	F	T	$0 = 0\cdot 1$
F	F	F	$0 = 0\cdot 0$

Table 6.7-4

6.7.2.3 *Proof by enumeration for equational representation of disjunction*

Define the function $\check{r}(q;\varphi,\sigma)$ which represents the equation corresponding to disjunction $(\vee)$. Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \rightarrow 1$ and $F \rightarrow 0$.

$\varphi(q)$	$\vee$	$\sigma(q)$	$\check{r}(q;\varphi,\sigma) = \check{\varphi}(q) + \check{\sigma}(q) - \check{\varphi}(q)\cdot\check{\sigma}(q)$
T	T	T	$1 = 1 + 1 - 1\cdot 1$
T	T	F	$1 = 1 + 0 - 1\cdot 0$
F	T	T	$1 = 0 + 1 - 0\cdot 1$
F	F	F	$0 = 0 + 0 - 0\cdot 0$

Table 6.7-5

6.7.2.4 *Proof by enumeration for equational representation of implication*

Define the function $\check{r}(q;\varphi,\sigma)$ which represents the equation corresponding to disjunction $(\Longrightarrow)$. First note the equivalence of

$$\varphi(q) \Longrightarrow \sigma(q) \text{ and } \sim\varphi(q) \vee \sigma(q).$$

(6.7-7)

Verify by enumeration the correspondence of the mathematical equation values corresponding to the mapping $T \to 1$ and $F \to 0$.

$\varphi(q)$	$\sim\varphi(q)$	$\vee$	$\sigma(q)$	$\check{r}(q;\varphi,\sigma) = 1 - \check{\varphi}(q) + \check{\varphi}(q)\cdot\check{\sigma}(q)$
T	F	T	T	$1 = 1 - 1 + 1 \cdot 1$
T	F	F	F	$0 = 1 - 1 + 1 \cdot 0$
F	T	T	T	$1 = 1 - 0 + 0 \cdot 1$
F	T	T	F	$1 = 1 - 0 + 0 \cdot 0$

Table 6.7-6

6.7.2.5 Proof for equational representation of tail recursion

Tail recursion is propositionally defined as

$$\varphi(q) \Leftarrow \varphi_1(q) \wedge \varphi_2(q) \wedge \cdots \wedge \varphi_{k-1}(q) \wedge \varphi(q)$$

(6.7-8)

where q represents the current state and the subscript k indicates the kth rule. To develop an equational representation of the recursive formulation, first define the general function $\tilde{\varphi}(n,q)$ where n represents the n^{th} iteration of the tail recursion and $\tilde{\varphi}(n,q)$ is the logical consequent. Then rewrite the above formulation using the recursive step, where the kth rule is instantiated for the nth time,

$$\tilde{\varphi}_1(n,q) \wedge \tilde{\varphi}_2(n,q) \wedge \cdots \wedge \tilde{\varphi}_{k-1}(n,q) \wedge \tilde{\varphi}(n-1,q) \Longrightarrow \tilde{\varphi}(n,q).$$

(6.7-9)

Define

$$\begin{aligned}\tilde{\sigma}(n,q) &\cong \tilde{\varphi}_1(n,q) \wedge \tilde{\varphi}_2(n,q) \wedge \cdots \wedge \tilde{\varphi}_{k-1}(n,q) \\ \tilde{r}(n-1,q) &\cong \tilde{\sigma}(n,q) \wedge \tilde{\varphi}(n-1,q)\end{aligned}$$

(6.7-10)

Then the tail recursion is rewritable as

$$\begin{aligned}&\tilde{\sigma}(n,q) \wedge \tilde{\varphi}(n-1,q) \Longrightarrow \tilde{\varphi}(n,q) \\ &\tilde{r}(n-1,q) \Longrightarrow \tilde{\varphi}(n,q).\end{aligned}$$

(6.7-11)

According to the equational representation of implication, let

$$\check{h}(n-1,q) = 1 - \check{\tilde{\sigma}}(n,q)\cdot\check{\tilde{\varphi}}(n-1,q) + \check{\tilde{\sigma}}(n,q)\cdot\check{\tilde{\varphi}}(n-1,q)\cdot\check{\tilde{\varphi}}(n,q).$$

(6.7-12)

Since by definition $\hat{\tilde{r}}(n-1,q) = \hat{\tilde{\sigma}}(n,q)\cdot\hat{\tilde{\varphi}}(n-1,q)$. Then

$$\check{\tilde{\varphi}}(n,q) = \frac{\hat{h}(n-1,q) + \hat{\tilde{\sigma}}(n,q)\cdot\hat{\tilde{\varphi}}(n-1,q) - 1}{\hat{\tilde{\sigma}}(n,q)\cdot\hat{\tilde{\varphi}}(n-1,q)}$$

(6.7-13)

with boundary condition $n = 0$.

6.7.3 Converting rules based system of inference to the problem of constrained minimization

TBD

6.7.3.1 Converting rules to constraints

The preceding discussion has established an algorithm for converting rules of the form

$$h(q) \Longleftarrow \varphi_1(q) \wedge \varphi_2(q) \wedge \cdots \wedge \varphi_m(q)$$

(6.7-14)

to constraints of the form

$$\check{h}(q) = \check{\varphi}_1(q)\cdot\check{\varphi}_2(q)\cdot\ \cdots\ \cdot\check{\varphi}_m(q).$$

(6.7-15)

6.8 Decision Element (DE)

A diagram of the Decision Element (DE) architecture is shown in Figure 6.8-1. It is composed of seven elements:

1. List of external repositories (LER)
2. Programmable search engine (PSE)
3. Internal heterogeneous database (IHDB)
4. Inference engine (IE)
5. Inference rule base (IRB)
6. API / user interface (UI)
7. Network interface (NI)

A functional description of these elements follows. In Figure 6.8-1, the IRB is internal to the DE, whereas the IHBD is external to the DE but accessible (as shown in Figure 6.1-1).

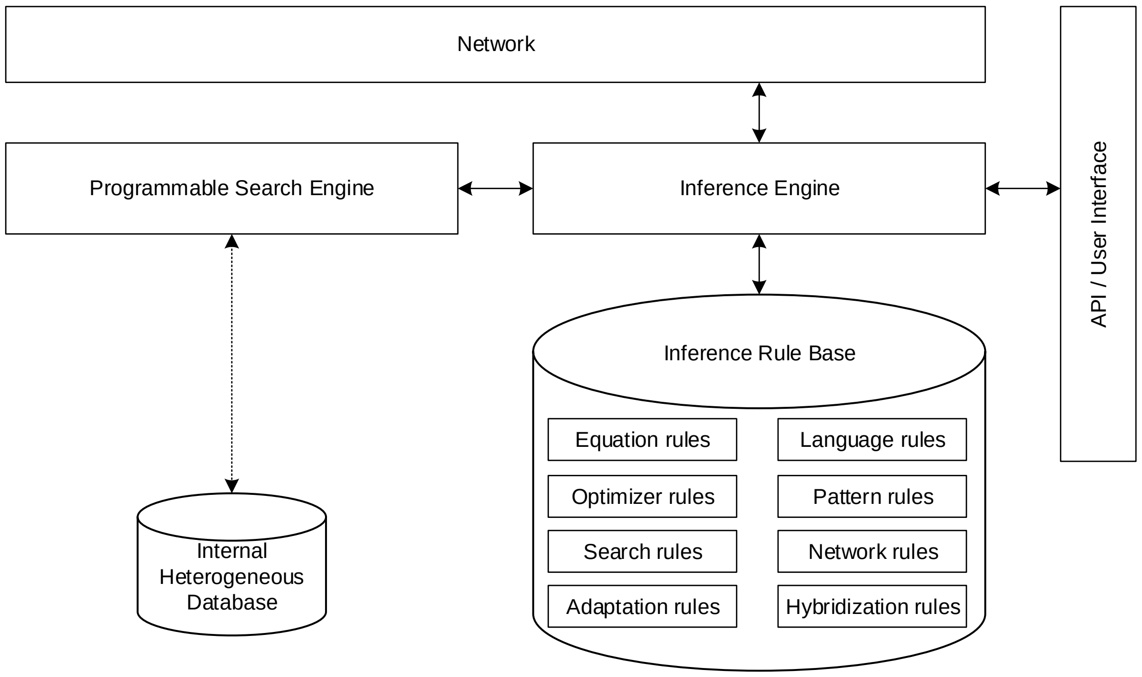


Fig. 6.8-2. Decision Element Architecture.

6.8.1 List of External Repositories (LER)

A DE has a *List of External Repositories* (LER). Each entry in an LER includes 1) a *protocol*, 2) a *heading sub-list*, and 3) a *translation grammar*. Each protocol entry prescribes the access procedure to the corresponding external knowledge repository. Each heading sub-list entry contains a summary of the knowledge contents of the corresponding repository. Finally, each translation grammar entry provides a procedure for converting knowledge elements of the corresponding repository in to the rule representation in the IHDB of the DE.

6.8.2 Programmable search engine (PSE)

The programmable search engine implements a standard hashing algorithm for detecting active rules as a function of the current instantiation of the variables in a *variable buffer* (VB) of the IE, and the contents of the *active rule buffer* (ARB). The VB contains the variables that form part of the query and all additional variables incorporated to this buffer during the *inference process* (IP). The VB includes all relevant data from the EKB beneficial to perform the query. The IP is described below. The ARB contains all the rules that are currently active in the IP. See the CDI Implementation Document for more description of the PSE.

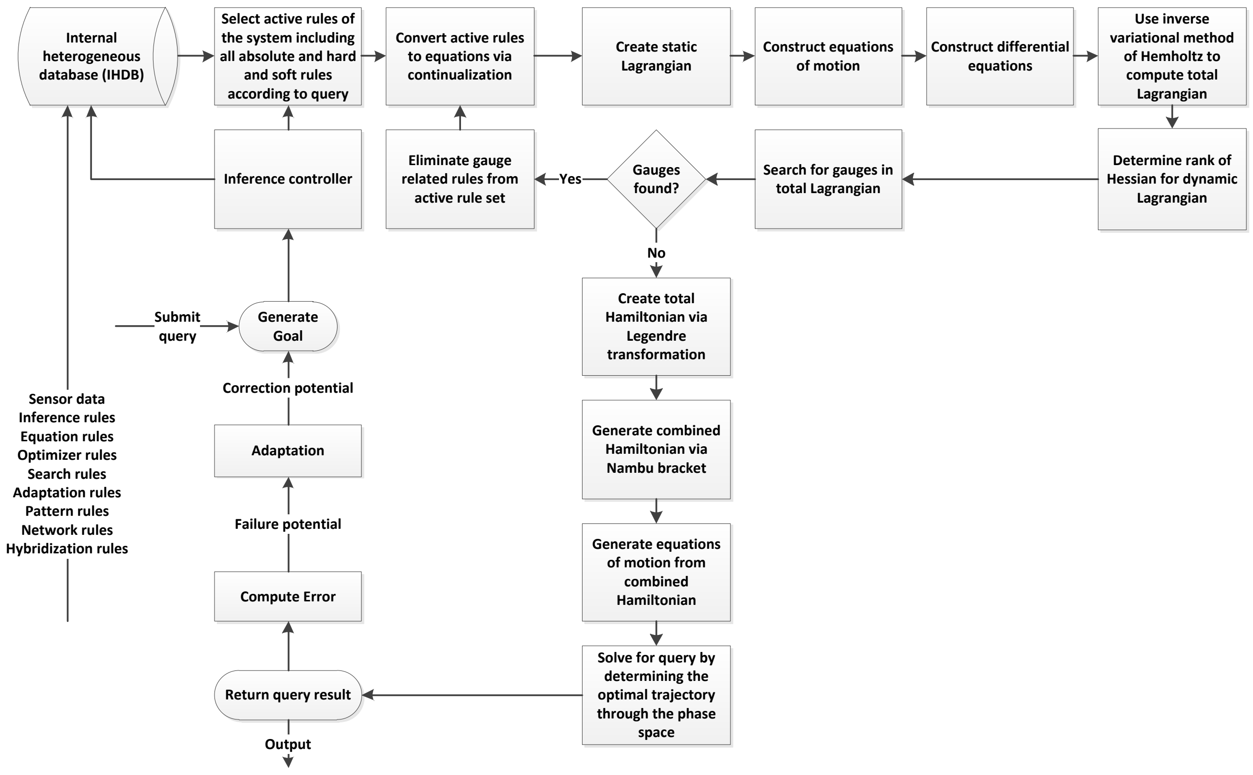
The search hashing algorithm is characterized by the *search rules* in the Inference Rule Base (see Figure 6.8-1).

6.8.3 Internal heterogeneous database (IHDB)

The IHDB is the repository of the *application clauses* associated with the DE. These encode the domain of knowledge characterizing the expertise of the DE. For example in a medical application, a decision element may deal with expertise on heart illnesses, and the corresponding clauses might encode diagnoses and treatments for these diseases.

6.8.4 Inference engine (IE)

The IE encodes an algorithm, the IP, for assigning values to the variables appearing in the query. The IP is summarized in the block diagram of Fig. 6.8-3.



Internal heterogeneous database (IHDB)
Select active rules of the system including all absolute and hard and soft rules according to query
Convert active rules to equations via continualization
Create static Lagrangian
Construct equations of motion
Construct differential equations
Use inverse variational method of Hemholtz to compute total Lagrangian
Determine rank of Hessian for dynamic Lagrangian
Search for gauges in total Lagrangian
Gauges found?
Yes
No
Eliminate gauge related rules from active rule set
Inference controller
Submit query
Generate Goal
Correction potential
Adaptation
Failure potential
Compute Error
Return query result
Output
Sensor data
Inference rules
Equation rules
Optimizer rules
Search rules
Adaptation rules
Pattern rules
Network rules
Hybridization rules
Create total Hamiltonian via Legendre transformation
Generate combined Hamiltonian via Nambu bracket
Generate equations of motion from combined Hamiltonian
Solve for query by determining the optimal trajectory through the phase space

Fig. 6.8-3. Inference Process

6.8.5 Inference rule types

The DE incorporates *inference rules* (IR) that are a collection of rules for transforming and inferring instantiations of the goal. These rules provide the Inference Engine with directives for processing database rules to give a satisfactory instantiation to a given query or to request additional information so that a satisfactory instantiation can be generated. They are shown in Fig. 6.8-3 in the IRB, and are organized according to their functionality, as follows.

6.8.5.1 Equation rules

These rules include the formal rules for inference. This includes all rules for natural language modeling from first principles.

6.8.5.2 Optimizer rules

These rules include rules for finding the interior point in optimization.

6.8.5.3 Search rules

These rules include rules for identifying the nature of insufficient potential. The goal is to apply these rules to acquire additional information required to satisfy the optimization goal.

6.8.5.4 Adaptation rules

Adaptation rules are used to update the soft rules to relax them further to reduce the complexity and constraints of the optimization problem. The adaptation also serves to update the search rules to improve information acquisition.

6.8.5.5 Language rules and Pattern rules

These rules embody the machine learning models.

6.8.5.6 Network rules

These rules define how information is distributed over the network and what information is available from which resources.

6.8.5.7 Hybridization rules

The rules define how other rules may be combined.

6.8.6 User interface (UI)

The UI provides the utilities for entering queries, pragma rules, displaying query answers, status and for general interaction with the IE.

6.8.7 Network interface (NI)

The NI provides a generic mechanism for interacting with other DEs via a procedure termed *companionship*. The companionship procedure implements the active coupling for the cooperation of the DEs in query resolution. This procedure is not hierarchical and implements a Pareto Agreement set strategy as the mechanism for CDI.

6.9 Query Language Interface (QLI)

Once a query is submitted (see the CDI Implementation Document), it is transformed into equational form, instantiated with the active rules, and a Hamiltonian is created for use in the optimization. The action is denoted $M(t)$, and its time derivative is denoted $\dot{M}(t)$.

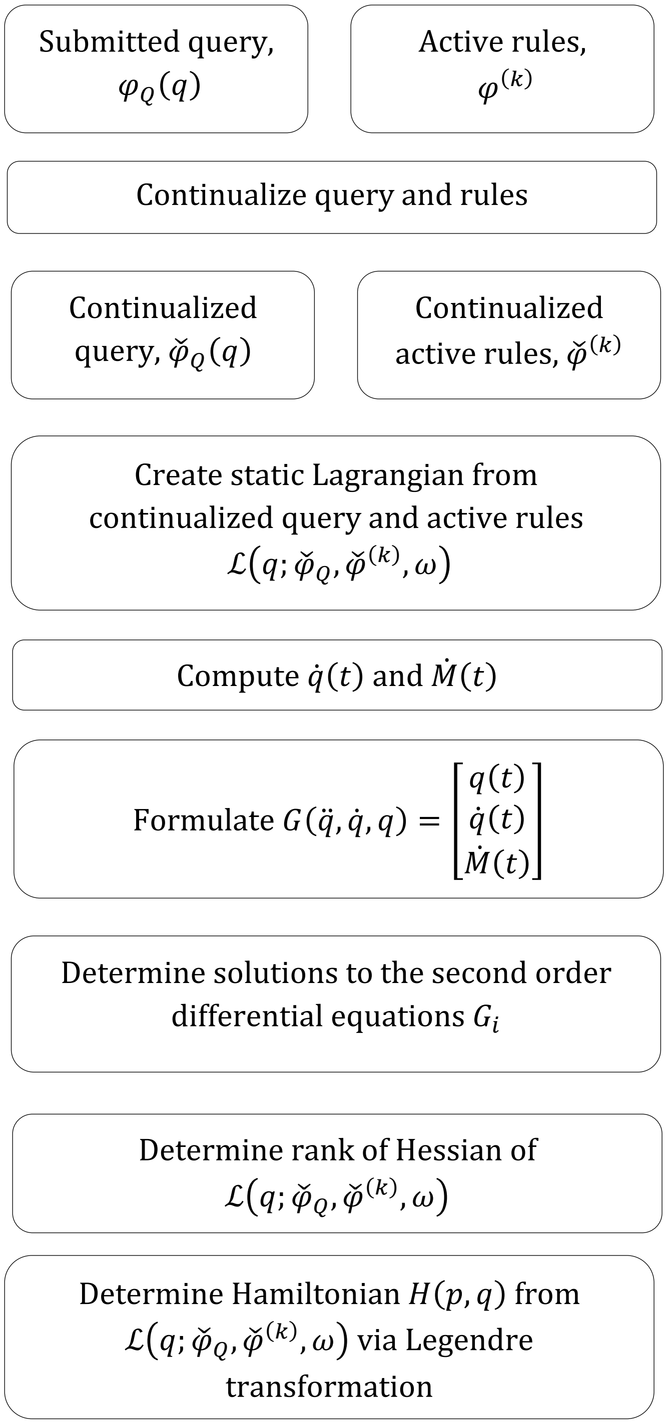

Submitted query, $\varphi_Q(q)$
Active rules, $\varphi^{(k)}$
Continualize query and rules
Continualized query, $\breve{\varphi}_Q(q)$
Continualized active rules, $\breve{\varphi}^{(k)}$
Create static Lagrangian from continualized query and active rules $\mathcal{L}(q; \breve{\varphi}_Q, \breve{\varphi}^{(k)}, \omega)$
Compute $\dot{q}(t)$ and $\dot{M}(t)$
Formulate $G(\ddot{q}, \dot{q}, q) = \begin{bmatrix} q(t) \\ \dot{q}(t) \\ \dot{M}(t) \end{bmatrix}$
Determine solutions to the second order differential equations G_i
Determine rank of Hessian of $\mathcal{L}(q; \breve{\varphi}_Q, \breve{\varphi}^{(k)}, \omega)$
Determine Hamiltonian $H(p, q)$ from $\mathcal{L}(q; \breve{\varphi}_Q, \breve{\varphi}^{(k)}, \omega)$ via Legendre transformation

Fig. 6.9-1. Process for developing Hamiltonian from a query and a set of active rules.

6.10 Minimization Function Generator (MFG) and Process for Determining Active Constraints

The minimization function generator converts a query to a minimization function. Again, we assume without loss of generality the entire set of canonical coordinates q is an argument to any proposition φ_i. In practice, we may further assume it is possible to apply the particular required coordinates as need to the proposition or function in question. Then let $\varphi^{(k)}$be the set of propositions associated with DE_k in the context of query Q. These propositions are composed of the proposition associated with the query $\varphi_Q(q)$, and other propositions $\varphi_i(q)$, comprising the constraints of the system. The proposition $\varphi_Q(q)$ associated with a given query Q can be converted to an equation $\breve{\varphi}_Q(q)$. Queries that are satisfiable specify a set.

$$\{q|\varphi_Q(q) \longleftarrow T\}$$

(6.10-1)

Similarly, a satisfied query represented as an equation is also a set

$$\{q|\breve{\varphi}_Q(q) = 1\}.$$

(6.10-2)

Relaxing the values that $\breve{\varphi}_Q(\cdot)$ can take to include the unit interval so that soft rules are incorporated yields the following constrained optimization expression. Let $J(q) = \left(\breve{\varphi}_Q(q) - 1\right)^2$ be the quadratic criterion that favors a value close to one (representing truth). Then the optimization problem is specified as,

$$\min_q J(q)$$

(6.10-3)

subject to:

1. $\breve{\varphi}_Q(q) \leq 1$
2. $\breve{\varphi}_Q(q) \geq 0$
3. A knowledge base on the set $\{\breve{\varphi}_1(q), \ldots, \breve{\varphi}_n(q), \ldots, \breve{\varphi}_{n+s}(q)\} \subseteq \breve{\varphi}^{(k)}$ which represents a further set of active constraints specific to the problem:
   a. $\breve{\varphi}_i(q) \geq 0$ for $1 \leq i \leq n$,
   b. $\breve{\varphi}_i(q) \leq 1$ or, equivalently $-(\breve{\varphi}_i(q) - 1) \geq 0$ for $1 \leq i \leq n$,

c. and, in the case of absolute and hard rules,
$\breve{\varphi}_l(q)\big(1-\breve{\varphi}_l(q)\big)=0$ for $n<l\leq n+s$.

Introduce the indicator functions

$$V_{\breve{\varphi}_i}^{-}=\begin{cases}0 & \breve{\varphi}_i(q)\geq 0\\ \infty & \breve{\varphi}_i(q)<0\end{cases}$$

(6.10-4)

and

$$V_{\breve{\varphi}_i}^{+}=\begin{cases}0 & 1-\breve{\varphi}_i(q)\geq 0\\ \infty & 1-\breve{\varphi}_i(q)<0\end{cases}$$

(6.10-5)

which yields the two logarithmic barrier functions

$$\breve{V}_{\breve{\varphi}_i}^{-}=-\log\big(\breve{\varphi}_i(q)\big)$$

(6.10-6)

and

$$\breve{V}_{\breve{\varphi}_i}^{+}=-\log\big(1-\breve{\varphi}_i(q)\big).$$

(6.10-7)

According to the method of Lagrange multipliers, combine this with the equality constraints to form the static Lagrangian function

$$\begin{aligned}\mathcal{L}&\left(q;\breve{\varphi}_Q,\breve{\varphi}^{(k)},\omega_1^{(+)},\dots,\omega_n^{(+)},\omega_{n+1}^{(-)},\dots,\omega_{2n}^{(-)},\omega_{2n+1}^{(\lambda)},\dots,\omega_{2n+s}^{(\lambda)},\omega_{2n+s+1}^{(Q)},\omega_{2n+s+2}^{(Q)}\right)\\ &=\breve{\varphi}_Q(q)+\sum_{i=1}^{n}\left[\omega_i^{(+)}\breve{V}_{\breve{\varphi}_i}^{+}+\omega_{n+i}^{(-)}\breve{V}_{\breve{\varphi}_i}^{-}\right]+\sum_{l=1}^{s}\omega_{2n+l}^{(\lambda)}\breve{\varphi}_l(q)\big(1-\breve{\varphi}_l(q)\big)\\ &-\omega_{2n+s+1}^{(Q)}\log\left(\breve{\varphi}_Q(q)\right)-\omega_{2n+s+2}^{(Q)}\log\left(1-\breve{\varphi}_Q(q)\right),\end{aligned}$$

(6.10-8)

the roots of which can be found using a formulation of Newton-Raphson. Since $\mathcal{L}$ here includes absolute, hard and soft rules we may call it the total static Lagrangian for DE_k and refer to it as $\mathcal{L}_k^{(T)}$.

6.11 Construct equations of motion

In a separate document

6.12 Query Response Engine (QRE) which includes Process for Constructing Differential Equations

6.12.1 Application of Newton-Raphson

Consider a continuous analog of the independent variables of $\mathcal{L}(\cdot)$

$$q = q(t) = \begin{bmatrix} q^{(1)}(t) \\ \vdots \\ q^{(v)}(t) \end{bmatrix}$$

(6.12-1)

where each of the v total independent variables of $\mathcal{L}(\cdot)$ is mapped to its corresponding position in $q(t)$, the column vector that is represented with a lower-case q. To reiterate, the independent variable t refers $algorithmic\ time$ as opposed to physical time which may also be represented in the system. The corresponding unconstrained optimization goal can be written as

$$\min_{q_1,\dots,q_v} \mathcal{L}\left(q^{(1)}(t),\dots,q^{(v)}(t)\right)$$

(6.12-2)

so that $\nabla L(q)$

$$\nabla \mathcal{L}\big(q(t)\big) = \begin{bmatrix} \dfrac{\partial \mathcal{L}}{\partial q^{(1)}} \\ \vdots \\ \dfrac{\partial \mathcal{L}}{\partial q^{(v)}} \end{bmatrix} = \begin{bmatrix} \nabla \mathcal{L}_1 \\ \vdots \\ \nabla \mathcal{L}_v \end{bmatrix} = 0,$$

(6.12-3)

with positive definite Hessian matrix

$$\nabla^2 \mathcal{L}\big(q(t)\big) = \begin{bmatrix} \dfrac{\partial \mathcal{L}}{\partial q^{(1)}\partial q^{(1)}} & \cdots & \dfrac{\partial \mathcal{L}}{\partial q^{(1)}\partial q^{(v)}} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial \mathcal{L}}{\partial q^{(v)}\partial q^{(1)}} & \cdots & \dfrac{\partial \mathcal{L}}{\partial q^{(v)}\partial q^{(v)}} \end{bmatrix} = \begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{1v} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{v1} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} => 0.$$

(6.12-4)

Write the recursion for Newton's method

$$q_{(k+1)}(t) = q_{(k)}(t) - \left(\nabla^2 \mathcal{L}\left(q_{(k)}(t)\right)\right)^{-1} \nabla \mathcal{L}\left(q_{(k)}(t)\right).$$

(6.12-5)

This is equivalently rewritten

$$\frac{q_{(k+1)}(t) - q_{(k)}(t)}{\delta} = -\frac{1}{\delta}\left(\nabla^2 \mathcal{L}\left(q_{(k)}(t)\right)\right)^{-1} \nabla \mathcal{L}\left(q_{(k)}(t)\right).$$

(6.12-6)

Via continualization we approximate the derivative

$$\dot{q}(t) = \frac{dq(t)}{dt} = -\left(\nabla^2 \mathcal{L}\big(q(t)\big)\right)^{-1} \nabla \mathcal{L}\big(q(t)\big).$$

(6.12-7)

6.12.2 Translation of inverted matrix

Consider M, an invertible and positive definite matrix. Then we make the following provable assertions.

1. $A^T A$ is symmetric.
2. $-\mathrm{A}^\mathrm{T}\mathrm{A}$ has negative eigenvalues.

Define

$$\frac{dM(t)}{dt} = -A^T A M(t) + A^T$$

(6.12-8)

Then as $t \to \infty$, $M(t) \to A^{-1} = \nabla^2 \mathcal{L}\left(q_{(k)}(t)\right)^{-1}$. Using (6.12-3) and (6.12-4) approximate $\dot{q}(t)$ by rewriting the derivative in the context of $M(t)$. This yields the following two equations.

$$\dot{q}(t) = -M(t)\nabla L\big(q(t)\big) = \begin{bmatrix} m_{11} & \cdots & m_{1v} \\ \vdots & \ddots & \vdots \\ m_{v1} & \cdots & m_{vv} \end{bmatrix} \begin{bmatrix} \nabla \mathcal{L}_1 \\ \vdots \\ \nabla \mathcal{L}_v \end{bmatrix} = \begin{bmatrix} m_{11}\nabla \mathcal{L}_1 + \cdots + m_{1v}\nabla \mathcal{L}_v \\ \vdots \\ m_{v1}\nabla \mathcal{L}_1 + \cdots + m_{vv}\nabla \mathcal{L}_v \end{bmatrix}$$

(6.12-9)

$$\frac{dM(t)}{dt} = -\left(\nabla^2 \mathcal{L}\big(q(t)\big)\right)^T \left(\nabla^2 \mathcal{L}\big(q(t)\big)\right) M(t) + \left(\nabla^2 \mathcal{L}\big(q(t)\big)\right)^T$$
$$= -\begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{v1} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{1v} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} \begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{1v} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{v1} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix} \begin{bmatrix} m_{11} & \cdots & m_{1v} \\ \vdots & \ddots & \vdots \\ m_{v1} & \cdots & m_{vv} \end{bmatrix} + \begin{bmatrix} \nabla \mathcal{L}_{11} & \cdots & \nabla \mathcal{L}_{v1} \\ \vdots & \ddots & \vdots \\ \nabla \mathcal{L}_{1v} & \cdots & \nabla \mathcal{L}_{vv} \end{bmatrix}$$

$$
= -\begin{bmatrix} \nabla\mathcal{L}_{11}^2 + \cdots + \nabla\mathcal{L}_{v1}^2 & \cdots & \nabla\mathcal{L}_{11}\nabla\mathcal{L}_{1v} + \cdots + \nabla\mathcal{L}_{v1}\nabla\mathcal{L}_{vv} \\ \vdots & \ddots & \vdots \\ \nabla\mathcal{L}_{11}\nabla\mathcal{L}_{1v} + \cdots + \nabla\mathcal{L}_{v1}\nabla\mathcal{L}_{vv} & \cdots & \nabla\mathcal{L}_{1v}^2 + \cdots + \nabla\mathcal{L}_{vv}^2 \end{bmatrix} \begin{bmatrix} m_{11} & \cdots & m_{1v} \\ \vdots & \ddots & \vdots \\ m_{v1} & \cdots & m_{vv} \end{bmatrix}
$$
$$
+ \begin{bmatrix} \nabla\mathcal{L}_{11} & \cdots & \nabla\mathcal{L}_{v1} \\ \vdots & \ddots & \vdots \\ \nabla\mathcal{L}_{1v} & \cdots & \nabla\mathcal{L}_{vv} \end{bmatrix}
$$
$$
= -\begin{bmatrix} \begin{matrix} (\nabla\mathcal{L}_{11}^2 + \cdots + \nabla\mathcal{L}_{v1}^2)m_{11} + \cdots \\ +(\nabla\mathcal{L}_{11}\nabla\mathcal{L}_{1v} + \cdots + \nabla\mathcal{L}_{v1}\nabla\mathcal{L}_{vv})m_{v1} \\ +\nabla\mathcal{L}_{11} \end{matrix} & \cdots & \begin{matrix} (\nabla\mathcal{L}_{11}^2 + \cdots + \nabla\mathcal{L}_{v1}^2)m_{1v} + \cdots \\ +(\nabla\mathcal{L}_{11}\nabla\mathcal{L}_{1v} + \cdots + \nabla\mathcal{L}_{v1}\nabla\mathcal{L}_{vv})m_{vv} \\ +\nabla\mathcal{L}_{v1} \end{matrix} \\ \vdots & \ddots & \vdots \\ \begin{matrix} (\nabla\mathcal{L}_{11}\nabla\mathcal{L}_{1v} + \cdots + \nabla\mathcal{L}_{v1}\nabla\mathcal{L}_{vv})m_{11} + \cdots \\ +(\nabla\mathcal{L}_{1v}^2 + \cdots + \nabla\mathcal{L}_{vv}^2)m_{v1} \\ +\nabla\mathcal{L}_{1v} \end{matrix} & \cdots & \begin{matrix} (\nabla\mathcal{L}_{11}\nabla\mathcal{L}_{1v} + \cdots + \nabla\mathcal{L}_{v1}\nabla\mathcal{L}_{vv})m_{1v} + \cdots \\ +(\nabla\mathcal{L}_{1v}^2 + \cdots + \nabla\mathcal{L}_{vv}^2)m_{vv} \\ +\nabla\mathcal{L}_{vv} \end{matrix} \end{bmatrix}
$$
$$
= \begin{bmatrix} \nabla m_{11} & \cdots & \nabla m_{1v} \\ \vdots & \ddots & \vdots \\ \nabla m_{v1} & \cdots & \nabla m_{vv} \end{bmatrix}
$$

(6.12-10)

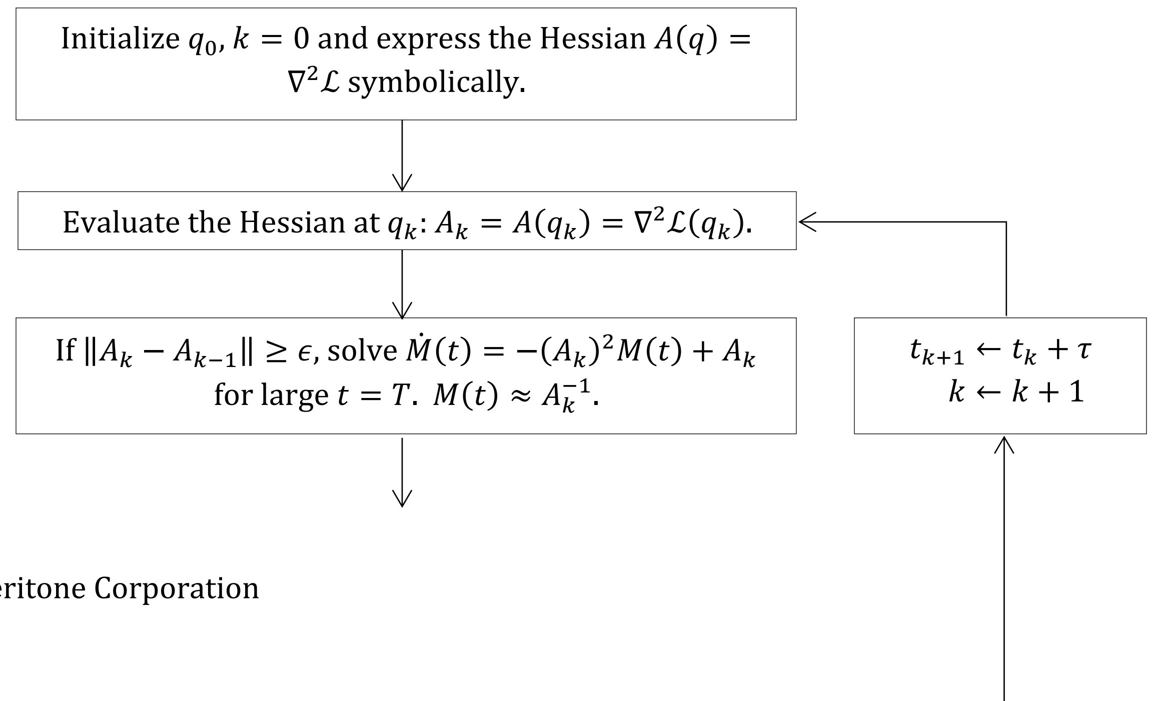
The approximation proceeds as follows:

1. Fix $M(0) = \nabla^2\mathcal{L}(q(0))$ and $= \nabla^2\mathcal{L}(q(t))$.
2. Use the variation of constants formula to solve

$$
M(T) = e^{-[\nabla^2\mathcal{L}(q(T))]^2 t} M(0) + \left[\int_0^T e^{-[\nabla^2\mathcal{L}(q(\tau))]^2 (T-\tau)}\, d\tau\right] \nabla^2\mathcal{L}(q(T))
$$

   applying the Magnus expansion to compute the integral.

The following figure documents the flow of computation.

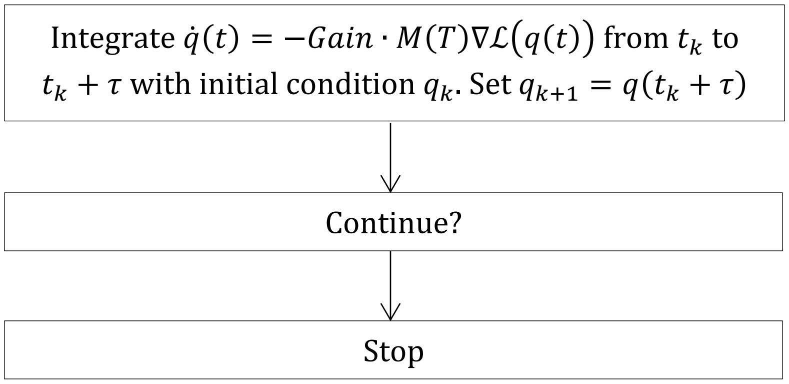


Initialize $q_0, k = 0$ and express the Hessian $A(q) = \nabla^2\mathcal{L}$ symbolically.

Evaluate the Hessian at q_k: $A_k = A(q_k) = \nabla^2\mathcal{L}(q_k)$.

If $\|A_k - A_{k-1}\| \geq \epsilon$, solve $\dot{M}(t) = -(A_k)^2 M(t) + A_k$ for large $t = T$. $M(t) \approx A_k^{-1}$.

$t_{k+1} \leftarrow t_k + \tau$
$k \leftarrow k + 1$


Integrate $\dot{q}(t) = -Gain \cdot M(T)\nabla\mathcal{L}(q(t))$ from t_k to $t_k + \tau$ with initial condition q_k. Set $q_{k+1} = q(t_k + \tau)$
Continue?
Stop

6.12.3 Process for determining dynamic Lagrangian via Hemholtz equations

Given

$$G_i(\ddot{q}, \dot{q}, q) = \sum_{j=1}^{n} W_{i,j}(\dot{q}, q)\,(\dot{q}, q)\ddot{q}^{(j)} + K_i(\dot{q}, q) = 0 \quad j = 1, \dots, n$$

(6.12-11)

If the three conditions

$$\begin{aligned}
&\frac{\partial G_i}{\partial \ddot{q}^{(j)}} = \frac{\partial G_j}{\partial \ddot{q}^{(i)}},\\
&\frac{\partial G_i}{\partial \dot{q}^{(j)}} + \frac{\partial G_j}{\partial \dot{q}^{(i)}} = \frac{d}{dt}\left(\frac{\partial G_i}{\partial \ddot{q}^{(j)}} + \frac{\partial G_j}{\partial \ddot{q}^{(i)}}\right),\\
&\frac{\partial G_i}{\partial q^{(j)}} - \frac{\partial G_j}{\partial q^{(i)}} = \frac{1}{2}\frac{d}{dt}\left(\frac{\partial G_i}{\partial \dot{q}^{(j)}} - \frac{\partial G_j}{\partial \dot{q}^{(i)}}\right),
\end{aligned}$$

(6.12-12)

with $i, j = 1, \dots, n$ hold, then

$$\sum_{j=1}^{n} \frac{\partial^2 L}{\partial \dot{q}^{(i)} \partial \dot{q}^{(j)}} \ddot{q}^{(j)} + \frac{\partial^2 L}{\partial q^{(j)} \partial \dot{q}^{(i)}} - \frac{\partial L}{\partial q^{(i)}} = G_i, \qquad i = 1, \dots, n$$

(6.12-13)

This is a second order, linear hyperbolic differential equation on the Lagrangian L. It can be solved efficiently by the method of characteristics.

Let

$$G(\ddot{q},\dot{q},q) = \begin{bmatrix} q(t) \\ \dot{q}(t) \\ \dot{M}(t) \end{bmatrix}$$
$$= \begin{bmatrix} q^{(1)} \\ \vdots \\ q^{(v)} \\ m_{11}\nabla\mathcal{L}_1 + \cdots + m_{1v}\nabla\mathcal{L}_v \\ \vdots \\ m_{v1}\nabla\mathcal{L}_1 + \cdots + m_{vv}\nabla\mathcal{L}_v \\ \nabla m_{11} \\ \vdots \\ \nabla m_{v1} \\ \vdots \\ \nabla m_{1v} \\ \vdots \\ \nabla m_{vv} \end{bmatrix}$$

(6.12-14)

6.12.4 Process for determining Hessian rank of dynamic Lagrangian

TBD

6.12.5 Converting the Lagrangian to the Hamiltonian via the Legendre transformation.

In our formulation the Lagrangian, $L_k^{(T)}(q,\dot{q};\omega)$, may be converted to the Hamiltonian using the Legendre transformation, so that

$$\begin{aligned} H_k^{(T)}(q,p;\omega) &= \frac{\partial L_k^{(T)}}{\partial \dot{q}}\dot{q} - L_k^{(T)}(q,\dot{q};\omega) \\ &= p^T\dot{q} - L_k^{(T)}(q,\dot{q};\omega) \end{aligned}$$

(6.12-15)

6.13 Pareto Multi-Criteria Optimization Engine (PMOE)

Consider the problem of determining the relaxed Pareto optimal solution to a given system query at a given time step. There are N decision elements, $k = 1, \dots, N$. A given decision element, DE_k, has the following associated parameters which are constituent to the ARB:

- A generalized set of coordinates relevant to DE_k, q.
- A generalized set of linearly independent momenta $\{p_a\}$ where the index a refers the linearly independent momenta selected from the canonical set p.
- A set of control parameters ω for hard a soft rules of the system, where $0 \le \omega_i \le 1$.

The ARB has the following components which determine the constraints of DE_k:

- The Hamiltonian which identifies the fundamental dynamics of the system of the system for the k'th decision element denoted

$$H_k^{(o)}(q, \{p_a\}).$$

(6.13-1)

- The summation of the *first class constraints* of the system, which is

$$\sum_i \omega_i f_i\left(q^{(i)}, \omega_i\right)$$

(6.13-2)

- The summation of the *second class constraints* of the system which is

$$\sum_i g_i\left(q^{(i)}, \omega_i\right)$$

(6.13-3)

- The Tellegen agent which is a function of the Hamiltonians of the absolute rules of the other $N - 1$ decision elements in the system

$$H_k^{(A)} = F_k^{(A)}\left(H_1^{(T)}, \dots, H_{k-1}^{(T)}, H_{k+1}^{(T)}, \dots, H_K^{(T)}\right)$$

(6.13-4)

- The total Hamiltonian of the system is denoted $H^{(T)}$.
- Approximations to the various Hamiltonian's are denoted$\widehat{H}_k^{(A)}$, $\widehat{H}^{(T)}$ and $\widehat{H}_k^{(o)}$ for the Tellegen, total, and DE-level Hamiltonians respectively.

6.13.1 System initialization

Determining the relaxed Pareto optimal point of the system is a process which includes:

1. Initialization of N decision elements.
2. Synchronization through companionship of each of the N decision elements with its respective Tellegen agent.

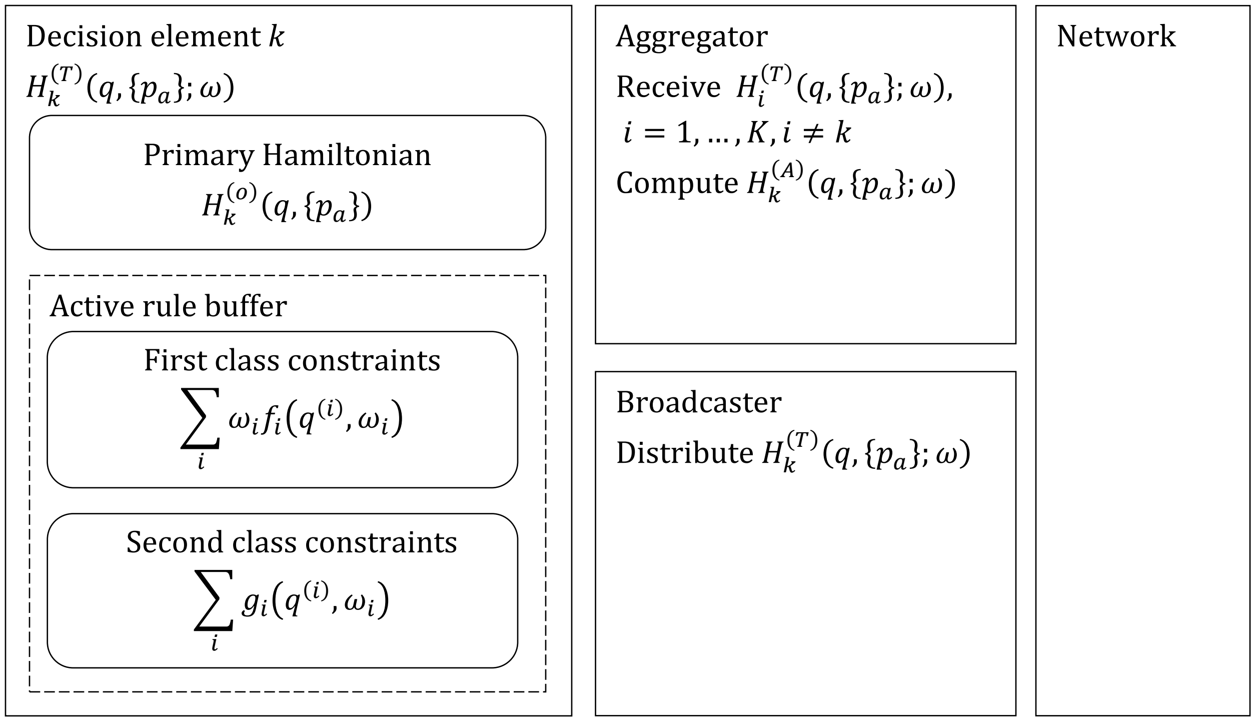

Decision element k
$H_k^{(T)}(q,\{p_a\};\omega)$
Primary Hamiltonian
$H_k^{(o)}(q,\{p_a\})$
Active rule buffer
First class constraints
$\sum_i \omega_i f_i\left(q^{(i)},\omega_i\right)$
Second class constraints
$\sum_i g_i\left(q^{(i)},\omega_i\right)$
Aggregator
Receive $H_i^{(T)}(q,\{p_a\};\omega)$, $i=1,\ldots,K, i\neq k$
Compute $H_k^{(A)}(q,\{p_a\};\omega)$
Broadcaster
Distribute $H_k^{(T)}(q,\{p_a\};\omega)$
Network

Fig. 6.13-1 shows the information components of the DE that are constituent to updating and being updated by the network at initialization.

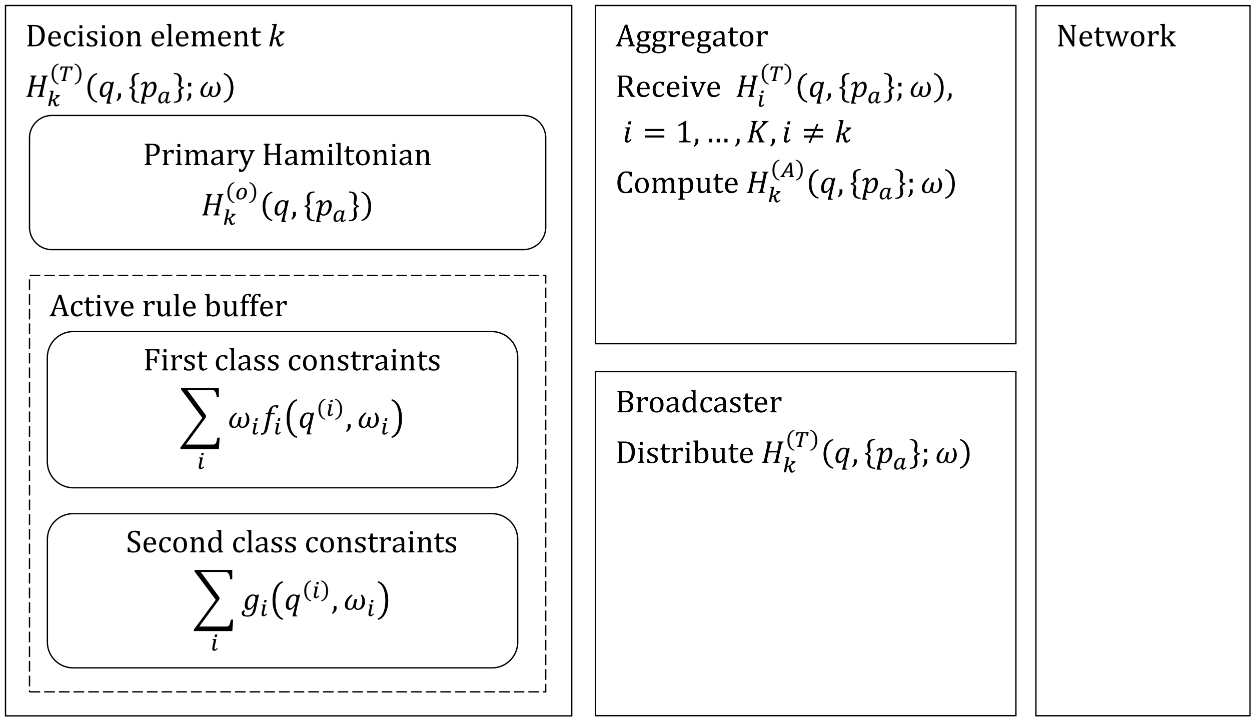

Decision element k
$H_k^{(T)}(q,\{p_a\};\omega)$
Primary Hamiltonian
$H_k^{(o)}(q,\{p_a\})$
Active rule buffer
First class constraints
$\sum_i \omega_i f_i(q^{(i)},\omega_i)$
Second class constraints
$\sum_i g_i(q^{(i)},\omega_i)$
Aggregator
Receive $H_i^{(T)}(q,\{p_a\};\omega)$,
$i = 1,\dots,K, i \neq k$
Compute $H_k^{(A)}(q,\{p_a\};\omega)$
Broadcaster
Distribute $H_k^{(T)}(q,\{p_a\};\omega)$
Network

Fig. 6.13-1

6.13.2 System operation

Fig. 6.13-2 shows how decision elements interact with the network, receive queries, and return results. In this example, the distributed system effectively implements an abstract classifier that has no real implementation. The DE's receive sensor data from the network which includes new available information which may benefit classification. The user submits a query that is received by a DE which then returns a result.

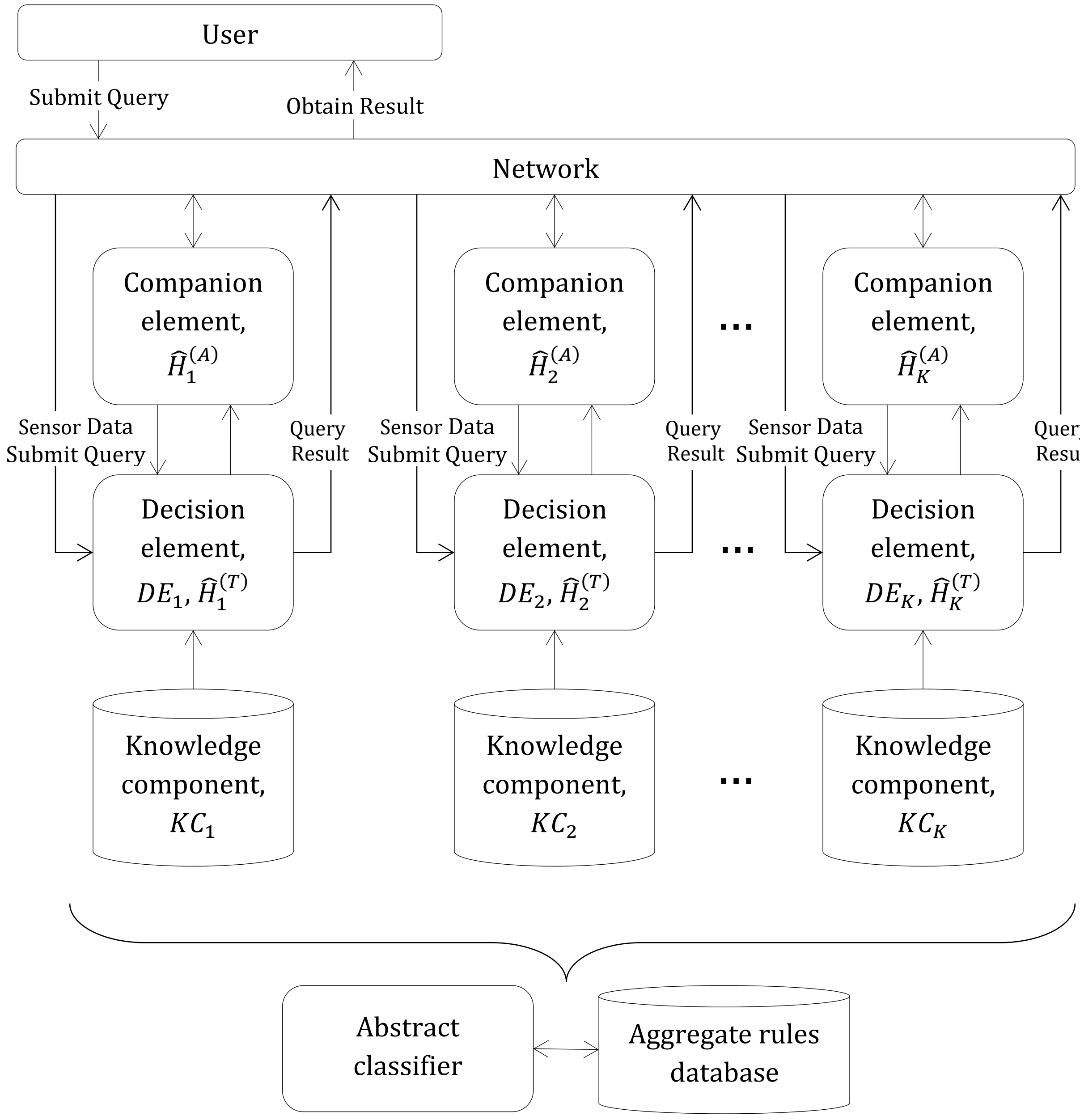

User
Submit Query
Obtain Result
Network
Companion element, $\widehat{H}_1^{(A)}$
Companion element, $\widehat{H}_2^{(A)}$
Companion element, $\widehat{H}_K^{(A)}$
Sensor Data
Submit Query
Query Result
Decision element, $DE_1, \widehat{H}_1^{(T)}$
Decision element, $DE_2, \widehat{H}_2^{(T)}$
Decision element, $DE_K, \widehat{H}_K^{(T)}$
Knowledge component, KC_1
Knowledge component, KC_2
Knowledge component, KC_K
Abstract classifier
Aggregate rules database

Fig. 6.13-2

Fig. 6.13-3 represents the iterative process of updating the Hamiltonian associated with DE_k.

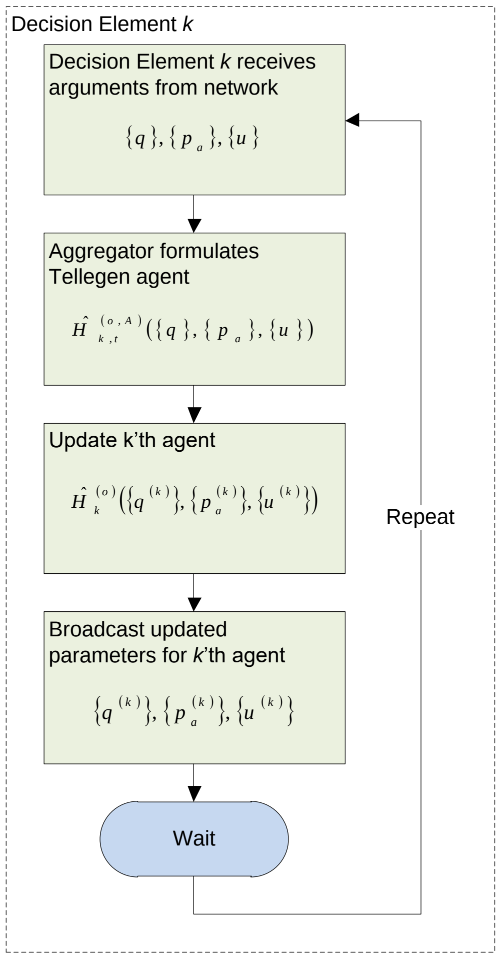

Decision Element k
Decision Element k receives arguments from network
$\{q\},\{p_a\},\{u\}$
Aggregator formulates Tellegen agent
$\hat{H}^{(o,A)}_{k,t}(\{q\},\{p_a\},\{u\})$
Update k'th agent
$\hat{H}^{(o)}_{k}(\{q^{(k)}\},\{p_a^{(k)}\},\{u^{(k)}\})$
Broadcast updated parameters for k'th agent
$\{q^{(k)}\},\{p_a^{(k)}\},\{u^{(k)}\}$
Wait
Repeat

Fig. 6.13-3

6.14 Gauge Systems in a Hamiltonian Domain

The time integral of the Lagrangian $L(q,\dot{q})$ is the action S_L defined as

$$S_L = \int_{t_1}^{t_2} L(q,\dot{q})dt$$

where $\dot{q} = \frac{dq(t)}{dt}$. The Lagrangian conditions for stationarity are first that

$$\frac{d}{dt}L_{\dot{q}^{(n)}} - L_{q^{(n)}} = 0$$

(6.14-1)

where $n = 1, \ldots, N$, $L_{\dot{q}^{(n)}} = \frac{\partial L}{\partial \dot{q}^{(n)}}$, and $L_{q^{(n)}} = \frac{\partial L}{\partial q^{(n)}}$. And, secondarily

$$\left[\sum_{n'=1}^{N} \ddot{q}^{(n')}\right] L_{\dot{q}^{(n)}\dot{q}^{(n)}} = L_{q^{(n)}} - \dot{q}^{(n)} L_{\dot{q}^{(n)}q^{(n)}}$$

(6.14-2)

where $\ddot{q}^{(n')} = \frac{d^2 q^{(n')}}{dt^2}$ and $L_{\dot{q}^{(n)}q^{(n)}} = \frac{\partial^2 L}{\partial \left(\dot{q}^{(n)}\right)^2}$. The generalized accelerations $\ddot{q}^{(n)}$ are immediately determined if $L_{\dot{q}^{(n)}\dot{q}^{(n)}}$ is invertible, or equivalently

$$\det\left(L_{\dot{q}^{(n)}\dot{q}^{(n)}}\right) \neq 0$$

(6.14-3)

for $i = 1, \ldots, N$. If for some n, $\det\left(L_{\dot{q}^{(n)}\dot{q}^{(n)}}\right) = 0$, the acceleration vector $\ddot{q}^{(n)}$ will not be uniquely determined.

The departing point for the Hamiltonian approach is the definition of conjugate momentum

$$p_n = L_{\dot{q}^{(n)}}$$

(6.14-4)

where $n = 1, \ldots, N$. We will see that (6.14-3) is the condition of non-invertibility of

$$L_{\dot{q}\dot{q}} = \begin{bmatrix} L_{\dot{q}^{(1)}\dot{q}^{(1)}} & \cdots & L_{\dot{q}^{(1)}\dot{q}^{(N)}} \\ \vdots & \iddots & \vdots \\ L_{\dot{q}^{(N)}\dot{q}^{(1)}} & \cdots & L_{\dot{q}^{(N)}\dot{q}^{(N)}} \end{bmatrix}$$

of the velocities of the functions of the coordinates q and momenta p. In other words, in this case, the momenta defined in (6.14-4) are not all independent. Define the relations that follow from (6.14-4) as

$$\phi_m(q, p)$$

(6.14-5)

where $m = 1, \ldots, M$. Write (6.14-4) in vector notation as

$$p = L_{\dot{q}}(q, \dot{q}).$$

Then compatibility demands

$$\phi_m\left(q, L_{\dot{q}}(q, \dot{q})\right) = 0$$

is an identity with $m = 1, \dots , M$.

Relations specified in (6.14-5) are called *primary constraints*. For simplicity let's assume that $rank(L_{\dot{q}\dot{q}})$ is constant throughout the phase space, $(q, \dot{q})$, so that (6.14-5) defines a submanifold smoothly embedded in the phase space. This manifold is known as the primary constraint surface.

Let

$$rank(L_{\dot{q}\dot{q}}) = N - M'$$

(6.14-6)

Then there are M' independent constraints among (6.14-5) and the primary constraint surface is a phase space submanifold of dimension $2N - M'$.

We do not assume that all the constraints are linearly independent so that

$$M' \leq M.$$

(6.14-7)

It follows from (6.14-5) that the inverse transformation from the p's to the q's is multivalued. That is, given q, p that satisfies (6.14-5), the inverse image $(q, \dot{q})$ that satisfies

$$p = \left(\frac{\partial L}{\partial \dot{q}}\right)^T$$

(6.14-8)

is not unique, since (6.14-8) defines a map from a $2N$-dimensional manifold $(q, \dot{q})$ to the smaller $(2N - M')$-dimensional manifold. Thus the inverse image of the points of (6.14-5) form a manifold of dimension M'.

6.14.1 Conditions on the Constraint Function

There exist many equivalent ways to represent a given surface by means of equations of the form of (6.14-5). For example the surface $p_1 = 0$ can be represented equivalently by $p_1^2 = 0$, $\sqrt{|p_1|} = 0$, or redundantly by $p_1 = 0$ and $p_1^2 = 0$. To use the Hamiltonian formalism, it is necessary to impose some restrictions which the regularity conditions for the constraints.

6.14.1.1 Regularity Conditions

1. The $(2N - M')$-dimensional constraint surface $\phi_m(q, p)$ should be covered of open region: in each region the constraints can be split into independent constraints

$$\{\phi_{m'} | m' = 1, \dots, M'\}.$$

Their Jacobian matrix

$$\left\{\frac{\partial\phi_{m'}}{\partial p_n, q^{(n)}}\right\} = \begin{bmatrix} \frac{\partial\phi_1}{\partial p_1, q^{(1)}} & \cdots & \frac{\partial\phi_1}{\partial p_n, q^{(n)}} \\ \vdots & \ddots & \vdots \\ \frac{\partial\phi_{m'}}{\partial p_1, q^{(1)}} & \cdots & \frac{\partial\phi_{m'}}{\partial p_n, q^{(n)}} \end{bmatrix}$$

with $m' = 1, \dots, M'$ and $n = 1, \dots, N$, is of rank M'.

The dependent constraints $\phi_m,\ m = M' + 1, \dots, M$ of the other $\phi_{m'} = 0 \Longrightarrow \phi_m = 0$. Alternatively the condition on the Jacobian.

2. The function $\phi_{m'}$ can be taken locally as the first M' coordinates of a new regular system in the vicinity of the constraint surface or the differentials $d\phi_1, \dots, d\phi_{M'}$ are locally linearly independent:

$$d\phi_1 \wedge \dots \wedge d\phi_{M'} \neq 0$$

(6.14-9)

3. The variations $\delta\phi_{m'}$ are of order ϵ for arbitrary variations $\delta q^{(i)}, \delta p_i$ of order ϵ (Dirac's approach).

Theorem 6.14.1. If a smooth, phase space function G vanishes on $\{\phi_m = 0\}$ then

$$G = \sum_{m=1}^{M} g^{(m)}\phi_m$$

(6.14-10)

Proof: (local proof). Set $\phi_{m'},\ m' = 1, \dots, M'$ as coordinates $(y_{m'}, x_\alpha)$ with $y_{m'} = \phi_{m'}$. In these coordinates $G(0, x) = 0$ and

$$\begin{aligned} G(y,x) &= \int_0^1 \frac{d}{dt} G(ty,x)dt \\ &= \sum_{m'=1}^{M'} y_{m'} \int_0^1 \frac{\partial}{\partial y_{m'}} G(ty,x)dt \\ &= \sum_{m'=1}^{M'} g^{(m')}(y,x)\phi_{m'}(y,x) \end{aligned}$$

with

$$g^{(m')}(y,x) = \int_0^1 \frac{\partial}{\partial y_{m'}} G(ty,x)dt.$$

(6.14-11)

Theorem 6.14.2. If the sum $\sum\left(\lambda^{(n)}\delta q^{(n)} + \mu_n \delta p_n\right) = 0$ for arbitrary variations $\delta q^{(i)}, \delta p_i$ tangent to the constraint surface $\{\phi_m(q,p) = 0 | m = 1, \dots, M\}$, then

$$\lambda^{(n)} = \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}}$$

(6.14-12)

$$\mu_n = \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n}$$

(6.14-13)

Proof. The dimension of $\{\phi_m\}$ is $2N - M'$. Thus the variations at a point (p,q) forms a $2N - M'$ dimensional space

$$\sum_{n=1}^{N}\left(\lambda^{(n)}\delta q^{(n)} + \mu_n \delta p_n\right) = 0$$

(6.14-14)

By the singularity assumption, there exists exactly M' solutions to (6.14-14). Clearly, the gradients $\left\{\frac{\partial \phi_{m'}}{\partial q^{(n)}}\right\}$ and $\left\{\frac{\partial \phi_{m'}}{\partial p_n}\right\}$ are linearly independent. They are the basis for solutions to (6.14-14).

Note that in the presence of redundant constraints, the functions $u^{(m)}$ exist but are not unique.

6.14.1.2 Canonical Hamiltonian

The Hamiltonian in canonical coordinates is

$$H(q,p) = \sum_{n=1}^{N} \dot{q}^{(n)} p_n - L(q,\dot{q})$$

(6.14-15)

The rate $\dot{q}$ enters through the combination through conjugate momenta defined for each coordinate

$$p_n(q,\dot{q}) = L_{\dot{q}^{(n)}}(q,\dot{q})$$

(6.14-16)

This remarkable property is essential for the Hamiltonian approach. It is verified by evaluating the change δH involved by arbitrary independent variations of position and velocities.

$$\begin{aligned}\delta H &= \sum_{n=1}^{N}\left(\dot{q}^{(n)}\delta p_n + \delta\dot{q}^{(n)} p_n\right) - \delta L \\ &= \sum_{n=1}^{N}\left(\dot{q}^{(n)}\delta p_n + \delta\dot{q}^{(n)} p_n\right) - \sum_{n=1}^{N}\left(L_{q^{(n)}}\delta q^{(n)} + L_{\dot{q}^{(n)}}\delta\dot{q}^{(n)}\right)\end{aligned}$$

(6.14-17)

Utilizing (6.14-16) in (6.14-17) yields

$$\delta H = \sum_{n=1}^{N}\left(\dot{q}^{(n)}\delta p_n - L_{q^{(n)}}\delta q^{(n)}\right)$$

(6.14-18)

The Hamiltonian defined by (6.14-15) is not unique as a function of p, q. This can be inferred from (6.14-18) by noticing that $\{\delta p_n | n = 1, \dots, N\}$ are not all independent. They are restricted to preserve the primary constraints $\phi_m \approx 0$ which are identities when the p's are expressed as functions of q's via (6.14-16).

Using the definition of the differential in several variables applied to $\delta H = \delta H(\{q^{(n)}\}, \{p_n\})$, (6.14-18) can be rewritten

$$\sum_{n=1}^{N}\left(\frac{\partial H}{\partial q^{(n)}}\delta q^{(n)} + \frac{\partial H}{\partial p_n}\delta p_n\right) = \sum_{n=1}^{N}\left(\dot{q}^{(n)}\delta p_n - \delta q^{(n)}\frac{\partial L}{\partial q^{(n)}}\right)$$

or

$$\sum_{n=1}^{N}\left(\frac{\partial H}{\partial q^{(n)}} + \frac{\partial L}{\partial q^{(n)}}\right)\delta q^{(n)} + \sum_{n=1}^{N}\left(\frac{\partial H}{\partial p_n} - \dot{q}^{(n)}\right)\delta p_n = 0$$

(6.14-19)

From theorem 2 we then conclude for each n that.

$$\frac{\partial H}{\partial q^{(n)}} + \frac{\partial L}{\partial q^{(n)}} = \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}}$$

and

$$\frac{\partial H}{\partial p_n} - \dot{q}^{(n)} = \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n}.$$

(6.14-20)

So for each n:

$$\dot{q}^{(n)} = \frac{\partial H}{\partial p_n} + \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n}, \qquad n = 1, \dots, N$$

(6.14-21)

and

$$-\frac{\partial L}{\partial q^{(n)}} = \frac{\partial H}{\partial q^{(n)}} + \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial q^{(n)}}, \qquad n = 1, \dots, N.$$

(6.14-22)

Note that if the constraints are independent, the vectors $\sum_{n=1}^{N} \frac{\partial \phi_m}{\partial p_n}, m = 1, \dots, M$ are also independent because of the regularity conditions (this is proved later). Hence no two sets of $\{u^{(m)} | m = 1, \dots, M\}$ can yield the same velocities via (6.14-21).

Thus, using

$$\dot{q}^{(n)} = \frac{\partial H}{\partial p_n} + \sum_{m=1}^{M} u^{(m)}(q, \dot{q}) \frac{\partial \phi_m}{\partial p_n}\big(q, p(q, \dot{q})\big)$$

we can find $u^{(m)}(p, \dot{q})$. If we define the transformation from $(q, \dot{q})$ to the manifold $\{\phi_m(q,p) = 0 | m = 1, \dots, M\}$, from $q, \dot{q}, u \to q, p, u$ by

$$\begin{aligned} &q = q, && n = 1, \dots, N \\ &p_n = L_{q^{(n)}}(q, \dot{q}), && n = 1, \dots, N - M' \\ &u^{(m)} = u^{(m)}(q, \dot{q}), && m = 1, \dots, M' \end{aligned}$$

We see that this transformation is invertible since one has from $q, p, u \to q, \dot{q}, u$

$$q = q$$

$$\dot{q}^{(n)} = \frac{\partial H}{\partial p_n} + \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n}$$
$$\phi_m(q,p) = 0$$

Thus invertibility of the Legendre transformation when

$$\det(L_{\dot{q}\dot{q}}) = 0$$

can be regained at the prices of adding extra variables.

6.14.2 Action Principle of the Hamiltonian Form

With (6.14-21) and (6.14-22) we can rewrite (6.14-1) in the equivalent Hamiltonian form

$$\dot{q}^{(n)} = \frac{\partial H}{\partial p_n} + \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n}$$
$$\dot{p}_n = -\frac{\partial H}{\partial p_n} - \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n}$$
$$\phi_m(q,p) = 0, \quad m = 1, \dots, M'$$

(6.14-23)

The Hamiltonian Equations (6.14-23) can be derived from the following variational principle:

$$\delta \int_{t_1}^{t_2} \left[\sum_{n=1}^{N} q^{(n)} p_n - H - \sum_{m=1}^{M} u^{(m)} \phi_m \right] = 0$$

(6.14-24)

for arbitrary variations of $\delta q^{(n)}$, δp_n, and $\delta u^{(m)}$ subject to

$$\delta q(t_1) = \delta q(t_2) = 0$$

where the $u^{(m)}$ appear now as Lagrange multipliers enforcing the primary constraints $\phi_m(q,p) = 0, \;\; m = 1, \dots, M$.

Let $F(p,q)$ be an arbitrary function of the canonical variables, then

$$\frac{dF}{dt} = \sum_{n=1}^{N} \frac{\partial F}{\partial q^{(n)}} \dot{q}_n + \sum_{n=1}^{N} \frac{\partial F}{\partial p_n} \dot{p}_n$$

$$
\begin{aligned}
&= \sum_{n=1}^{N} \frac{\partial F}{\partial q^{(n)}} \left[\frac{\partial H}{\partial p_n} + \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n} \right] + \sum_{n=1}^{N} \frac{\partial F}{\partial p_n} \left[-\frac{\partial H}{\partial p_n} - \sum_{m=1}^{M} u^{(m)} \frac{\partial \phi_m}{\partial p_n} \right] \\
&= [F, H] + \sum_{m=1}^{M} u^{(m)} [F, \phi_m]
\end{aligned}
$$

(6.14-25)

The equation (6.14-25) introduces the new binary operator [·,·] which is the Poisson bracket and has the form

$$
\begin{aligned}
[F, G] &= \sum_{n=1}^{N} \left[\frac{\partial F}{\partial q^{(n)}} \frac{\partial G}{\partial p_n} + \frac{\partial F}{\partial p_n} \frac{\partial G}{\partial q^{(n)}} \right] \\
&= \sum_{n=1}^{N} \left[F_{q^{(n)}} G_{p_n} + F_{p_n} G_{q^{(n)}} \right]
\end{aligned}
$$

(6.14-26)

6.14.3 Secondary Constraints

The basic consistency condition is that the primary constraints be preserved in time. So for

$$
F(p, q) = \phi_m(q, p)
$$

we should have that $\dot{\phi}_m = 0. \{\phi_m(q, p) = 0\}$. So this means

$$
[\phi_m, H] + \sum_{m'=1}^{M'} u^{(m')} [\phi_m, \phi_{m'}] = 0
$$

(6.14-27)

This equation can either reduce to a relation independent of the $u^{(m')}$, or, it may impose a restriction on the u's.

$$
u = -\{[\phi_m, \phi_{m'}]\}[\phi_m, H](q, p)
$$

(6.14-28)

In the case (6.14-27) is independent of the u's (6.14-27) is called a secondary constraint. The fundamental difference of secondary constraints with respect to primary constraints is that primary constraints is that primary constraints are the consequence of the definition (6.14-8) while secondary constraints depend on the dynamics.

If $X(q, p) = 0$ is an external constraint, we most impose a compatibility condition

$$[X,H] + \sum_{m=1}^{M'} u^{(m)}[X,\phi_m] = 0$$

(6.14-29)

Next we need to test whether this constraint:

$$\Phi(p,q) = [X,H] + \sum_{m=1}^{M'} u^{(m)}[X,\phi_m] = 0$$

(6.14-30)

(6.14-31)

Implies new secondary constraints or whether it only restricts the u's. After the process is finished we are left with a number of secondary constraints which will be denoted by

$$\phi_k = 0, \qquad k = M+1, \dots, M+K$$

where K is the total number of secondary constraints. In general, it will be useful to denote all the constraints (primary and secondary) in a uniform way as

$$\phi_j(q,p) = 0, \qquad j = 1, \dots, M+K = J$$

(6.14-32)

We make the same regularity assumptions on the full set of constraints.

6.14.4 Weak and Strong Equations

Equation (6.14-32) can be written as

$$\phi_j(\cdot) \approx 0$$

(6.14-33)

To emphasize, the quantity ϕ_j is numerically restricted to be zero but does not vanish throughout the space. What this is means is that ϕ_j has non-zero Poisson brackets with the canonical variables.

Let F, G be functions that coincide on the manifold $\{\phi_j \approx 0 | j = 1, \dots, J\}$ are said the be weakly equal and denoted by $F \approx G$. On the other hand, an equation that holds throughout the entire phase space and not just on the submanifold $\{\phi_j \approx 0\}$ is called strong. Hence, by theorem 1

$$F \approx G \Leftrightarrow F - G = \sum_{j=1}^{J} c^{(j)}(p,q)\phi_j$$

(6.14-34)

6.14.5 Restrictions on the Lagrange Multipliers

Assume that we have found a complete set of constraints

$$\{\phi_j \approx 0 \big| j = 1, \dots, J\}$$

(6.14-35)

$$[\phi_j, H] + \sum_{m=1}^{M} u^{(m)}[\phi_j, \phi_m] \approx 0$$

(6.14-36)

We consider (6.14-36) as a set of non-homogeneous linear equations with $M \le J$ unknowns with coefficients that are functions of the q's and p's.

The general solution of (6.14-36) for each j is of the form

$$u^{(m)} = U^{(m)} + V^{(m)}, \quad m = 1, \dots, M$$

(6.14-37)

with $V^{(m)}$ the solution of the homogeneous equation

$$\sum_{m=1}^{M} V^{(m)}[\phi_j, \phi_m] \approx 0$$

(6.14-38)

The most general solution of (6.14-38) is a linear combination of linearly independent solutions of $V_\alpha^{(m)}$ where $\alpha = 1, \dots, A$ with $A \le M$. Under the assumption that the matrix

$$\begin{bmatrix} [\phi_1, \phi_1] & \cdots & [\phi_1, \phi_M] \\ \vdots & \ddots & \vdots \\ [\phi_J, \phi_1] & \cdots & [\phi_J, \phi_M] \end{bmatrix}$$

(6.14-39)

is of constant rank, the number of independent solutions A is the same for all p, q. Thus the general solution to (6.14-36) can be written as

$$u^{(m)} \approx U^{(m)} + \sum_{\alpha=1}^{A} v^{(\alpha)} V_{\alpha}^{(m)}, \quad m = 1, \dots, M$$

(6.14-40)

6.14.6 Irreducible and Reducible Cases

If the equations $\{\phi_j = 0 | j = 1, \dots, J\}$ are not independent, one says that the constraints are reducible. The system is irreducible when the constraints are independent. However the separation of constraints into dependent and independent ones might be difficult to perform. It also may disturb invariance properties under some important symmetry. In some cases it may be impossible to separate irreducible from irreducible contexts. Reducible cases arise for example when the dynamical coordinates include p-form gauge fields.

Any irreducible set of constraints can always be replaced by a reducible set by introducing constraints ??? of the ones already at hand. The formalism should be invariant under such replacements.

6.14.7 Total Hamiltonian

We now discuss details of the dynamic equation (6.14-25)

$$\dot{F} \approx \left[F, H' + \sum_{\alpha=1}^{A} v^{(\alpha)} \phi_{\alpha} \right]$$

(6.14-41)

where from (6.14-40)

$$H' = H + \sum_{m=1}^{M} U^{(m)} \phi_m$$

and

$$\phi_{\alpha} = \sum_{m=1}^{M} V_{\alpha}^{(m)} \phi_m, \qquad \alpha = 1, \dots, A$$

(6.14-42)

This is the result of theorem 3 (see below).

Theorem 3.

$$\left[F, \sum_{m=1}^{M} U^{(m)} \phi_m\right] \simeq \sum_{m=1}^{M} U^{(m)} [F, \phi_m]$$

(6.14-43)

$$\left[F, \sum_{\alpha=1}^{A} V_\alpha^{(m)} \phi_m\right] \simeq \sum_{\alpha=1}^{A} V_\alpha^{(m)} [F, \phi_m]$$

(6.14-44)

Proof.

$$\begin{aligned}\left[F, \sum_{m=1}^{M} U^{(m)} \phi_m\right] &= \sum_{i=1}^{N} \left\{\frac{\partial F}{\partial q^{(i)}} \frac{\partial}{\partial p_i} \sum_{m=1}^{M} U^{(m)} \phi_m - \frac{\partial F}{\partial p_i} \frac{\partial}{\partial q^{(i)}} \sum_{m=1}^{M} U^{(m)} \phi_m\right\} \\ &= \sum_{i=1}^{N} \left\{\frac{\partial F}{\partial q^{(i)}} \left[\sum_{m=1}^{M} \frac{\partial U^{(m)}}{\partial p_i} \phi_m + \sum_{m=1}^{M} U^{(m)} \frac{\partial \phi_m}{\partial p_i}\right]\right\} \\ &\quad - \sum_{i=1}^{N} \left\{\frac{\partial F}{\partial p_i} \left[\sum_{m=1}^{M} \frac{\partial U^{(m)}}{\partial q^{(i)}} \phi_m + \sum_{m=1}^{M} U^{(m)} \frac{\partial \phi_m}{\partial q^{(i)}}\right]\right\} \\ &= \sum_{m=1}^{M} \left\{[F, U^{(m)}]\phi_m + U^{(m)}[F, \phi_m]\right\}\end{aligned}$$

So

$$\left[F, \sum_{m=1}^{M} U^{(m)} \phi_m\right] - \sum_{m=1}^{M} U^{(m)} [F, \phi_m] = \sum_{m=1}^{M} [F, U^{(m)}] \phi_m$$

(6.14-45)

and from (6.14-34) in (6.14-45), (6.14-43) follows. By a similar process we show (6.14-44). We now prove the validity of (6.14-41).

Theorem 4. Let $F(q,p)$ be a regular function, then $F(p,q)$ propagates in time according to the approximate equation (6.14-41).

Proof. From (6.14-25),

$$\frac{dF}{dt} = [F, H] + \sum_{m=1}^{M} u^{(m)} [F, \phi_m].$$

(6.14-46)

From (6.14-40) into (6.14-46) we obtain,

$$\frac{dF}{dt} \approx [F, H] + \sum_{m=1}^{M} \left\{ U^{(m)} + \sum_{\alpha=1}^{A} v^{(\alpha)} V_{\alpha}^{(m)} \right\} [F, \phi_m]$$

or

$$\frac{dF}{dt} \approx [F, H] + \sum_{m=1}^{M} U^{(m)} [F, \phi_m] + \sum_{m=1}^{M} \sum_{\alpha=1}^{A} v^{(\alpha)} V_{\alpha}^{(m)} [F, \phi_m]$$

(6.14-47)

Thus from (6.14-43) and (6.14-44) of theorem 3, we get

$$\begin{aligned} \frac{dF}{dt} &\approx [F, H] + \sum_{m=1}^{M} \left[F, U^{(m)} \phi_m\right] + \sum_{\alpha=1}^{A} v^{(\alpha)} \left[F, \sum_{m=1}^{M} V_{\alpha}^{(m)} \phi_m\right] \\ &\approx \left[F, H + \sum_{m=1}^{M} U^{(m)} \phi_m + \sum_{\alpha=1}^{A} v^{(\alpha)} \sum_{m=1}^{M} V_{\alpha}^{(m)} \phi_m\right] \\ &\approx \left[F, H' + \sum_{m=1}^{M} U^{(m)} \phi_m + \sum_{\alpha=1}^{A} v^{(\alpha)} \phi_{\alpha}\right] \end{aligned}$$

(6.14-48)

with

$$H' = H + \sum_{m=1}^{M} U^{(m)} \phi_m$$

(6.14-49)

$$\phi_{\alpha} = \sum_{m=1}^{M} V_{\alpha}^{(m)} \phi_m$$

(6.14-50)

Now define

$$H_T = H' + \sum_{\alpha=1}^{A} v^{(\alpha)} \phi_{\alpha}.$$

(6.14-51)

So we obtain

$$\frac{dF}{dt} \approx [F, H_T]$$

(6.14-52)

6.14.8 First and Second Class Functions

The distinction between primary and secondary constraints is of little importance. We now consider a **fundamental classification**. It depends on the concept of first class and second class functions.

Definition 1. A function $F(q, p)$ is said to be *first class* if its Poisson bracket with every constraint vanishes weakly, $[F, \phi_j] \approx 0, j = 1, \dots, J$. A function of the canonical variables that is not first class is called *second class.* Thus F is second class if $[F, \phi_k] \not\approx 0$ for at least one $k, k = 1, \dots, M$.

Theorem 5. If F and G are first class functions, then their Poisson bracket is also a first class function.

Proof: By Hypothesis,

$$[F, \phi_j] = \sum_{k=1}^{M} f_j^{(k)} \phi_k$$

(6.14-53)

$$[G, \phi_j] = \sum_{l=1}^{M} g_j^{(l)} \phi_l$$

(6.14-54)

Applying the Jacobi identity, we get

$$\begin{aligned}
\left[[F, G], \phi_j\right] &= \left[F, [G, \phi_j]\right] - \left[G, [F, \phi_j]\right] \\
&= \left[F, \sum_{l=1}^{M} g_j^{(l)} \phi_l\right] - \left[G, \sum_{k=1}^{M} f_j^{(k)} \phi_k\right] \\
&= \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \frac{\partial}{\partial p_i} \sum_{l=1}^{M} g_j^{(l)} \phi_l - \frac{\partial F}{\partial p_i} \frac{\partial}{\partial q^{(i)}} \sum_{l=1}^{M} g_j^{(l)} \phi_l \right\} \\
&\quad - \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \frac{\partial}{\partial p_n} \sum_{k=1}^{M} f_j^{(k)} \phi_k - \frac{\partial G}{\partial p_n} \frac{\partial}{\partial q^{(n)}} \sum_{k=1}^{M} f_j^{(k)} \phi_k \right\}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \sum_{l=1}^{M} \left\{ \frac{\partial g_j^{(l)}}{\partial p_i} \phi_l + g_j^{(l)} \frac{\partial \phi_l}{\partial p_i} \right\} - \frac{\partial F}{\partial p_i} \sum_{l=1}^{M} \left\{ \frac{\partial g_j^{(l)}}{\partial q^{(i)}} \phi_l + g_j^{(l)} \frac{\partial \phi_l}{\partial q^{(i)}} \right\} \right\} \\
&\quad - \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \sum_{k=1}^{M} \left\{ \frac{\partial f_j^{(k)}}{\partial p_n} \phi_k + f_j^{(k)} \frac{\partial \phi_k}{\partial p_n} \right\} - \frac{\partial G}{\partial p_n} \sum_{k=1}^{M} \left\{ \frac{\partial f_j^{(k)}}{\partial q^{(n)}} \phi_k + f_j^{(k)} \frac{\partial \phi_k}{\partial q^{(n)}} \right\} \right\} \\
&= \sum_{l=1}^{M} \left\{ \phi_l \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \frac{\partial g_j^{(l)}}{\partial p_i} - \frac{\partial F}{\partial p_i} \frac{\partial g_j^{(l)}}{\partial q^{(i)}} \right\} + g_j^{(l)} \sum_i \left\{ \frac{\partial F}{\partial q^{(i)}} \frac{\partial \phi_l}{\partial q^{(i)}} - \frac{\partial F}{\partial p_i} \frac{\partial \phi_l}{\partial p_i} \right\} \right\} \\
&\quad - \sum_{k=1}^{M} \left\{ \phi_k \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \frac{\partial f_j^{(k)}}{\partial p_n} - \frac{\partial G}{\partial p_n} \frac{\partial f_j^{(k)}}{\partial q^{(n)}} \right\} + f_j^{(k)} \sum_n \left\{ \frac{\partial G}{\partial q^{(n)}} \frac{\partial \phi_k}{\partial p_n} - \frac{\partial G}{\partial p_n} \frac{\partial \phi_k}{\partial q^{(n)}} \right\} \right\} \\
&= \sum_{l=1}^{M} \left\{ \phi_l \left[F, g_j^{(l)} \right] + g_j^{(l)} [F, \phi_l] \right\} - \sum_{k=1}^{M} \left\{ \phi_k \left[G, f_j^{(k)} \right] + f_j^{(k)} [G, \phi_k] \right\} \\
&= \sum_{l=1}^{M} \left[F, g_j^{(l)} \right] \phi_l - \sum_{k=1}^{M} \left[G, f_j^{(k)} \right] \phi_k + \sum_{l'=1}^{M} \left\{ \sum_{l=1}^{M} g_j^{(l)} f_l^{(l')} \right\} \phi_{l'} - \sum_{k'=1}^{M} \left\{ \sum_{k=1}^{M} f_j^{(k)} g_{k'}^{k} \right\} \phi_{k'} \\
&\approx 0
\end{aligned}
$$

We now use theorem 5 to show the following.

Theorem 6. H' defined by (6.14-49) and ϕ_α defined by (6.14-50) are first class functions.

Proof: This follows directly from (6.14-36) and (6.14-38).

We learn from theorem 6 that the total Hamiltonian defined by (6.14-51) is the sum of the first class Hamiltonian H' and the first class primary constraints ϕ_α multiplied by arbitrary coefficients.

6.14.9 First Class Constraints as Generators of Gauge Transformations

Gauge transformations are transformations that do not change the physical state.

The presence of arbitrary functions of time $v^{(\alpha)}$, $\alpha = 1, \dots, A$ in the total Hamiltonian, H_T (see (6.14-51)) imply that not all the q's and p's are observable given a set of q's and p's where the state of the physical system is uniquely determined. However the converse is not true: there is more than one set of values of the canonical variables that defines a state. To illustrate this, we see that if we give an initial set of values of physical state at time t, we expect the equations of motion to fully determine the state at other times. Thus any ambiguity in the value of the canonical variables at $t_2 \neq t_1$ should be irrelevant from the physical point of view.

6.15 A Derivation Example

We propose here an alternate formulation of Dirac's formalism.

6.15.1 Primary Constraints

Recall that the momenta, canonically conjugate to the generalized "coordinates" $q^{(j)}, j = 1, \dots, N$ is given by

$$p_j = \frac{\partial L(q, \dot{q})}{\partial \dot{q}^{(j)}}, \qquad j = 1, \dots, N.$$

(E -- 1)

For non-singular systems the equations in allows us to express $\dot{q}^{(j)}, j = 1, \dots, N$ in terms of the canonical variables,

$$\dot{q}^{(i)} = f_i(q, p), \qquad i = 1, \dots, N$$

(E -- 2)

By performing a Legendre transformation

$$H_c(p, q) = \sum_{i=1}^{N} p_i f(q, p) + L\big(q, f(p, q)\big)$$

We obtain the Hamiltonian of the system H_c. And from this function we obtain the standard equations of motion of the system.

$$\dot{q} = \frac{\partial H_c}{\partial p}$$
$$\dot{p} = -\frac{\partial H_c}{\partial q}$$

(E -- 3)

For (E -- 2) to be well-defined we need to have the Hessian W of satisfy

$$\det W \neq 0$$

(E -- 4)

In this case the accelerations $\ddot{q}^{(i)}$ are uniquely determined by the $q^{(j)}$ and $\dot{q}^{(i)}$.

When $\det W \neq 0$, the Hamiltonian equations of motion do not take the standard form, and we speak of a singular Lagrangian. For illustration purposes, consider a Lagrangian of the form

$$L(q,\dot{q}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} W_{ij}(q)\dot{q}^{(i)}\dot{q}^{(j)} + \sum_{i=1}^{N}\eta_i(q)\dot{q}^{(i)} - V(q)$$

(E -- 5)

with W a symmetric matrix. From (E -- 1), the canonical momentum for (E -- 5) is given by

$$p_i = \frac{1}{2}\sum_{j=1}^{N} W_{ij}(q)\dot{q}^{(j)} + \eta_i(q), \quad i = 1,\dots,n.$$

(E -- 6)

If W is singular of rank R_W, then it possesses $N - R_W$ eigenvectors with corresponding zero eigenvalues. Then for eigenvectors $v_j^{(\alpha)}$

$$\sum_{j=1}^{N} W_{ij}(q)v_j^{(\alpha)}(q) = 0, \quad \alpha = 1,\dots,N-R_W$$

So pre-multiplying (E -- 6) by $v_j^{(\alpha)}$ and summing over i we get

$$\begin{aligned}\sum_{i=1}^{N} v_i^{(\alpha)}(q)\,p_i &= \sum_{i=1}^{N}\left[\sum_{j=1}^{N}\left(v_i^{(\alpha)}(q)W_{ij}(q)\dot{q}^{(j)}\right) + v_i^{(\alpha)}(q)\eta_i(q)\right] \\ &= \sum_{i=1}^{N} v_i^{(\alpha)}(q)\eta_i(q)\,, \quad \alpha = 1,\dots,N-R_W\end{aligned}$$

So

$$\sum_{i=1}^{N} v_i^{(\alpha)}(q)\left(p_i - \eta_i(q)\right) = 0, \quad \alpha = 1,\dots,N-R_W.$$

(E -- 7)

Let $\{p_\alpha\}$, $\alpha = 1,\dots,N-R_W$, denote the linearly dependent elements of p. Let $\{p_a\}$, $a = 1,\dots,R_a$ be the momenta satisfying (E -- 1). Then the constraint equations are of the form

$$\sum_{\beta=1}^{N-R_W} M_{\alpha\beta}(q)p_\beta - F_\alpha(q,\{p_a\}) = 0, \quad \alpha = 1,\dots,N-R_W$$

(E -- 8)

$$M_{\alpha\beta}(q) = v_{\beta}^{(\alpha)}$$

and

$$F_{\alpha}(q, \{p_{\beta}\}) = \sum_{i=1}^{N} v_i^{(\alpha)}(q)\eta_i(q) + \sum_{b=1}^{R_W} v_b^{(\alpha)}(q)p_b$$

(E -- 9)

The matrix $\{M_{\alpha\beta}\}$ is necessarily invertible because otherwise M would possess eigenvectors with zero eigenvalues, implying existence of additional constraints.

Note that (E -- 8) can be written as

$$p_{\alpha} - g_{\alpha}(q, \{p_a\}) = 0, \quad \alpha = 1, \dots, N - R_W$$

with

$$g_{\alpha}(q, \{p_a\}) = \sum_{\beta=1}^{N-R_W} M_{\alpha\beta}^{-1} F_{\beta}(q, \{p_a\})$$

(E -- 10)

with $\dim\{p_a\} = R_W$. So we can define,

$$\phi_{\alpha}(q, p) = p_{\alpha} - g_{\alpha}(q, \{p_a\}) = 0, \quad \alpha = 1, \dots, N - R_W$$

(E -- 11)

In Dirac's terminology, constraints of the form of (E -- 11) are referred to as *primary constraints*. Although the derivation above is based on a Lagrangian, quadratic in the velocity terms, it is generally valid for Lagrangians which depend on q and $\dot{q}$ but not on higher derivatives.

Note: *Primary constraints follow exclusively from the definition of canonical momenta*.

The derivation above is valid for general Lagrangians and their Hessian. Let's assume $\{W_{ij}(q, \dot{q})\}$ is the Hessian of a given Lagrangian L. Let $\{W_{ab} | a, b = 1, \dots, R_W\}$ be the largest sub-matrix of $\{W_{ij}\}$ with suitable rearrangement if necessary. We then solve (E -- 1) for R_W velocities $\dot{q}^{(a)}$ in terms of $\{q^{(i)} | i = 1, \dots, n\}, \{p_a | a = 1, \dots, R_W\}$ and $\{q^{(\alpha)} | \alpha = 1, \dots, N - R_W\}$. That is

$$\dot{q}^{(a)} = f_a(q, \{p_b\}, \{\dot{q}^{(\beta)}\})$$

(E -- 12)

with $a, b = 1, \dots, R_W$ and $\beta = R_W + 1, \dots, N$.

Inserting these relations into (E -- 1), we get relations of the form

$$p_j = h_j\big(q, \{p_a\}, \{\dot{q}^{(\alpha)}\}\big)$$

(E -- 13)

with $a, j = 1, \dots, R_W$ and $\alpha = R_W + 1, \dots, N$. This relation reduces to an identity by construction. The remaining equations are of the form

$$p_\alpha = h_\alpha\big(q, \{p_a\}, \{\dot{q}^{(\beta)}\}\big)$$

(E -- 14)

with $\alpha = 1, \dots, N - R_W$. However, the right hand side cannot depend on $\{\dot{q}^{(\beta)}\}$ since otherwise we could express more velocities in terms of the momenta of the coordinates of the momenta and the remaining velocities.

6.16 Hamiltonian Equations of Motion for Constrained Systems

Theorem 6.16.1. In the space Γ_p define by $\Gamma_p = \{\phi_\alpha(p,q) | \alpha = 1, \dots, N - R_W\}$ where ϕ_α is defined as (E -- 11). The Hamiltonian is only a function of $\{q^{(i)} | i = 1, \dots, N\}$ and momenta $\{p_a | a = 1, \dots, R_W\}$ and does *not* depend on $\{\dot{q}^{(\alpha)} | \alpha = 1, \dots, N - R_W\}$

Proof. On Γ_p the Hamiltonian is given by

$$H_o = H_c|_{\Gamma_p} = \sum_{a=1}^{R_W} p_a f_a - \sum_{\alpha=1}^{N-R_W} g_\alpha \dot{q}^{(\alpha)} - L\big(q, \{f_b\}, \{\dot{q}^{(\beta)}\}\big)$$

(E -- 15)

where f_a, $a = 1, \dots, N - R_W$ is given by (E -- 12) and g_α, $\alpha = 1, \dots, R_W$ is given by (E -- 10). We want to show that H_o does not depend on $\dot{q}^{(\beta)}$, $\beta = 1, \dots, N - R_W$. We compute

$$\begin{aligned} \frac{\partial H_o}{\partial \dot{q}^{(\beta)}} &= \sum_{a=1}^{R_W} p_a \frac{\partial f_a}{\partial \dot{q}^{(\beta)}} - g_\beta - \sum_{a=1}^{R_W} \frac{\partial L}{\partial \dot{q}^{(a)}}\bigg|_{\dot{q}^{(a)} = f_a} \frac{\partial f_a}{\partial \dot{q}^{(\beta)}} - \frac{\partial L}{\partial \dot{q}^{(\beta)}}\bigg|_{\dot{q}^{(a)} = f_a} \\ &= \sum_{a=1}^{R_W} \left(p_a - \frac{\partial L}{\partial \dot{q}^{(a)}}\bigg|_{\dot{q}^{(a)} = f_a} \right) \frac{\partial f_a}{\partial \dot{q}^{(\beta)}} - g_\beta - \frac{\partial L}{\partial \dot{q}^{(\beta)}}\bigg|_{\dot{q}^{(a)} = f_a} \end{aligned}$$

(E -- 16)

Since by definition

$$p_a = \frac{\partial L}{\partial \dot{q}^{(a)}}, \quad a = 1, \dots, R_W$$

And from (E -- 11)

$$g_\beta = p_\beta = \frac{\partial L}{\partial \dot{q}^{(\beta)}}\bigg|_{\dot{q}_a = f_a}.$$

So

$$\frac{\partial H_o}{\partial \dot{q}^{(\beta)}} = 0, \quad \beta = 1, \dots, N - R_W.$$

(E -- 17)

and therefore

$$H_o\big(q, \{p_a\}, \{\dot{q}^{(a)}\}\big) = H_o(q, \{p_a\}).$$

Theorem 6.16.2. In the presence of primary constraints (E -- 11), the Hamilton equations of motion are given by

$$\begin{aligned} \dot{q}^{(i)} &= \frac{\partial H_o}{\partial p_i} + \sum_{\beta=1}^{N} \dot{q}^{(\beta)} \frac{\partial \phi_\beta}{\partial p_i}, \qquad i = 1, \dots, N \\ \dot{p}_i &= -\frac{\partial H_o}{\partial q^{(i)}} + \sum_{\beta=1}^{n} \dot{q}^{(\beta)} \frac{\partial \phi_\beta}{\partial q^{(i)}}, \quad i = 1, \dots, N \\ \phi_\alpha(p, q) &= 0, \qquad\qquad \alpha = 1, \dots, N - R_W \end{aligned}$$

(E -- 18)

where $\dot{q}^{(\beta)}$ are *a priori* underdetermined velocities.

Proof: From (E -- 15) we obtain and the application of **Theorem 6.16.1**

$$\begin{aligned} \frac{\partial H_o}{\partial p_a} &= f_a + \sum_{b=1}^{R_W} p_b \frac{\partial f_b}{\partial p_a} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_a} \dot{q}^{(\beta)} - \sum_{b=1}^{R_W} \frac{\partial L}{\partial \dot{q}^{(b)}} \frac{\partial f_b}{\partial p_a} \\ &= \dot{q}^{(a)} + \sum_{b=1}^{R_W} \left(p_b - \frac{\partial L}{\partial \dot{q}^{(b)}} \right) \frac{\partial f_b}{\partial p_a} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_a} \dot{q}^{(\beta)} \\ &= \dot{q}^{(a)} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_a} \dot{q}^{(\beta)} \end{aligned}$$

(E -- 19)

with $a = 1, \dots, n - R_W$. Further

$$\begin{aligned}
\frac{\partial H_o}{\partial q^{(i)}} &= \sum_{b=1}^{R_W} p_b \frac{\partial f_b}{\partial q^{(i)}} + \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}} - \frac{\partial L}{\partial q^{(i)}}\bigg|_{\dot{q}_a = f_a} - \sum_{b=1}^{R_W} \frac{\partial L}{\partial \dot{q}_b}\bigg|_{\dot{q}_b = f_b} \frac{\partial f_b}{\partial q^{(i)}} \\
&= \sum_{b=1}^{R_W} \left(p_b - \frac{\partial L}{\partial \dot{q}^{(b)}}\bigg|_{\dot{q}^{(b)} = f_b} \right) \frac{\partial f_b}{\partial q^{(i)}} + \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}} - \frac{\partial L}{\partial q^{(i)}}\bigg|_{\dot{q}^{(a)} = f_a} \\
&= \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}} - \frac{\partial L}{\partial q^{(i)}}\bigg|_{\dot{q}^{(a)} = f_a} \\
&= \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}^{(i)}} \right)\bigg|_{\dot{q}^{(a)} = f_a}
\end{aligned}$$

(E -- 20)

from (add reference).

$$\frac{\partial H_o}{\partial q^{(i)}} = -\dot{p}_i + \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}}$$

(E -- 21)

From (E -- 19) and (E -- 20) we get:

$$\begin{aligned}
\dot{q}^{(a)} &= \frac{\partial H_o}{\partial p_a} - \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_a} \dot{q}^{(\beta)}, \qquad a = 1, \dots, R_W \\
\dot{p}_i &= -\frac{\partial H_o}{\partial q^{(i)}} + \sum_{\beta=1}^{n-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}}, \qquad i = 1, \dots, N
\end{aligned}$$

(E -- 22)

Since $\frac{\partial H_o}{\partial p_\alpha} = 0$ and $\frac{\partial \phi_\beta}{\partial p_\alpha} = \delta\beta_\alpha$ we can supplement these equations with

$$\dot{q}^{(\alpha)} = \frac{\partial H_o}{\partial p_\alpha} - \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_\alpha} \dot{q}^{(\beta)}, \qquad \alpha = 1, \dots, N - R_W$$

(E -- 23)

So we can write

$$\dot{q}^{(i)} = \frac{\partial H_o}{\partial p_i} + \sum_{\beta=1}^{N-R_W} \frac{\partial g_\beta}{\partial p_i} \dot{q}^{(\beta)}, \qquad i = 1, \dots, N$$

$$\dot{p}_i = -\frac{\partial H_o}{\partial q^{(i)}} - \sum_{\beta=1}^{N-R_W} \dot{q}^{(\beta)} \frac{\partial g_\beta}{\partial q^{(i)}}, \quad i = 1, \dots, N$$

(E -- 24)

For consistency with (E -- 11) we should write

$$\dot{q}^{(\alpha)} = \frac{d}{dt} - g_\alpha(q, \{p_a\}), \qquad \alpha = 1, \dots, N - R_W$$

(E -- 25)

where $\dot{p}_\alpha$ is given by the right hand side of (E -- 22).

6.16.1 Streamlining the Hamiltonian equation of motion (EOM)

Definition 6.16-1. A function f is weakly equal to g denoted by $f \approx g$, if f and g are equal on the subspace defined by the primary constraints,

$$\phi_\beta = 0 \text{ when } f|_{\Gamma_p} = g|_{\Gamma_p}$$

and

$$f(q,p) \approx g(q,p) \Leftrightarrow f(q,p) = g(q,p) \text{ when } \{\phi_\alpha(q,p) = 0\}$$

Theorem 6.16.3. Assume f, g are defined over the entire space spanned by $\{q^{(i)}\}, \{p_i\}$. Then if

$$f(q,p)|_{\Gamma_p} = g(q,p)|_{\Gamma_p}$$

(E -- 26)

Then

$$\frac{\partial}{\partial q^{(i)}}\left(f - \sum_\beta \phi_\beta \frac{\partial f}{\partial p_\beta}\right) \simeq \frac{\partial}{\partial q^{(i)}}\left(h - \sum_\beta \phi_\beta \frac{\partial h}{\partial p_\beta}\right)$$

and

$$\frac{\partial}{\partial p_i}\left(f - \sum_\beta \phi_\beta \frac{\partial f}{\partial p_\beta}\right) \simeq \frac{\partial}{\partial p}\left(h - \sum_\beta \phi_\beta \frac{\partial h}{\partial p_\beta}\right)$$

(E -- 27)

for $i = 1, \ldots N$.

Proof: Consider the two functions $f\big(q, \{p_a\}, \{p_\beta\}\big)$ and $h\big(q, \{p_a\}, \{p_\beta\}\big)$. Using (E -- 11) and from the hypothesis of the theorem,

$$f(q, \{p_a\}, \{g_\alpha\}) = h(q, \{p_a\}, \{g_\alpha\})$$

(E -- 28)

Thus is follows

$$\left(\frac{\partial f}{\partial q^{(i)}} + \sum_a \frac{\partial f}{\partial p_a}\frac{\partial p_a}{\partial q^{(i)}} + \sum_\beta \frac{\partial f}{\partial p_\beta}\frac{\partial g_\beta}{\partial q^{(i)}}\right)_{\Gamma_p} = \left(\frac{\partial h}{\partial q^{(i)}} + \sum_a \frac{\partial h}{\partial p_a}\frac{\partial p_a}{\partial q^{(i)}} + \sum_\beta \frac{\partial h}{\partial p_\beta}\frac{\partial g_\beta}{\partial q^{(i)}}\right)_{\Gamma_p}$$

(E -- 29)

and

$$\left(\frac{\partial f}{\partial p_i} + \sum_{a\neq i} \frac{\partial f}{\partial p_a}\frac{\partial p_a}{\partial p_i} + \sum_\beta \frac{\partial f}{\partial p_\beta}\frac{\partial g_\beta}{\partial p_i}\right)_{\Gamma_\beta} = \left(\frac{\partial h}{\partial p_i} + \sum_{a\neq i} \frac{\partial h}{\partial p_a}\frac{\partial p_a}{\partial p_i} + \sum_\beta \frac{\partial h}{\partial p_\beta}\frac{\partial g_\beta}{\partial p_i}\right)_{\Gamma_\beta}$$

(E -- 30)

Note since $\phi_\alpha(q,p) = p_\alpha - g_\alpha(q, \{p_a\})$, we have

$$\frac{\partial g_\beta}{\partial q^{(i)}} = -\frac{\partial \phi_\beta(q,p)}{\partial q^{(i)}}$$

and

$$\frac{\partial g_\beta}{\partial p_i} = -\frac{\partial \phi_\beta(q,p)}{\partial p_i}$$

and

$$\partial \phi_\alpha(q,p) = 0$$

for $\alpha = 1, \ldots, N - R_W$. We have

$$\left(\frac{\partial f}{\partial q^{(i)}} - \sum_\beta \frac{\partial f}{\partial p_\beta}\frac{\partial \phi_\beta}{\partial q^{(i)}}\right)_{\Gamma_p} = \left(\frac{\partial h}{\partial q^{(i)}} - \sum_\beta \frac{\partial h}{\partial p_\beta}\frac{\partial \phi_\beta}{\partial q^{(i)}}\right)_{\Gamma_\beta}$$

which can be written as

$$\frac{\partial}{\partial q^{(i)}}\left(f - \sum_{\beta} \phi_\beta \frac{\partial f}{\partial p_\beta}\right) \simeq \frac{\partial}{\partial q^{(i)}}\left(h - \sum_{\beta} \phi_\beta \frac{\partial h}{\partial p_\beta}\right)$$

since $\phi_\beta \, {\partial^2 f}\big/{\partial p_\beta^2} = 0$ because $\phi_\beta = 0$. Similarly,

$$\frac{\partial}{\partial p_i}\left(f - \sum_{\beta} \phi_\beta \frac{\partial f}{\partial p_\beta}\right) \simeq \frac{\partial}{\partial p_i}\left(h - \sum_{\beta} \phi_\beta \frac{\partial h}{\partial p_\beta}\right)$$

Corrolary 6.16-1.

$$\dot{q}^{(i)} = \frac{\partial H}{\partial p_i} + \sum_{\beta} v^{(\beta)} \frac{\partial \phi_\beta}{\partial p_i}$$
$$\dot{p}_i = -\frac{\partial H}{\partial q^{(i)}} - \sum_{\beta} v^{(\beta)} \frac{\partial \phi_\beta}{\partial q^{(i)}}$$

for $i = 1, \dots, N$.

Proof. We consider two Hamiltonians $H(\{q^{(i)}\},\{p_i\})$ and $H_o(\{q^{(i)}\},\{p_a\})$. Define $H(\{q^{(i)}\},\{p_i\})$ as follows

$$H(\{q^{(i)}\},\{p_i\}) \approx H_o(\{q^{(i)}\},\{p_a\}).$$

Then using the result of Theorem 6.16.1, from (E -- 29) with $f = H$ and $h = H_o$

$$\frac{\partial H_o}{\partial q^{(i)}} \approx \frac{\partial}{\partial q^{(i)}}\left(H - \sum_{\beta=1}^{N-R_W} \phi_\beta \frac{\partial H}{\partial p_\beta}\right)$$

(E -- 31)

$$\frac{\partial H_o}{\partial p_i} \approx \frac{\partial}{\partial p_i}\left(H - \sum_{\beta=1}^{N-R_W} \phi_\beta \frac{\partial H}{\partial p_\beta}\right)$$

(E -- 32)

Using (E -- 31) and (E -- 32) in (E -- 24), we get

$$\dot{q}^{(i)} \approx \frac{\partial}{\partial p_i}\left(H - \sum_{\beta} \phi_\beta \frac{\partial H}{\partial p_\beta}\right) + \sum_{\beta} \dot{q}^{(\beta)} \frac{\partial \phi_\beta}{\partial p_i}$$

and

$$\dot{p}_i \approx -\frac{\partial}{\partial q^{(i)}}\left(H - \sum_{\beta} \phi_\beta \frac{\partial H}{\partial p_\beta}\right) - \sum_{\beta} \dot{q}^{(\beta)} \frac{\partial \phi_\beta}{\partial q^{(i)}}$$

or

$$\dot{q}^{(i)} \approx \frac{\partial}{\partial p_i}\left(H - \sum_{\beta} \phi_\beta \left(\frac{\partial H}{\partial p_\beta} - \dot{q}^{(\beta)}\right)\right)$$

and

$$\dot{p}_i \approx -\frac{\partial}{\partial q^{(i)}}\left(H - \sum_{\beta} \phi_\beta \left(\frac{\partial H}{\partial p_\beta} - \dot{q}^{(\beta)}\right)\right)$$

(E -- 33)

Define

$$v_\beta \equiv \dot{q}^{(\beta)} - \frac{\partial H}{\partial p_\beta}$$
$$H_T \equiv H + \sum_{\beta} v^{(\beta)} \phi_\beta$$

So (E -- 33) becomes

$$\dot{q}^{(i)} \approx \frac{\partial H_T}{\partial p_i}$$
$$\dot{p}_i \approx -\frac{\partial H_T}{\partial q^{(i)}}$$

(E -- 34)

6.17 Constrained Hamiltonian Systems

Local symmetries on a Lagrangian based model. Consider

$$q^{(i)} \longrightarrow q^{(i)}(t) + \delta q^{(i)}(t)$$

$$\dot{q}^{(i)} \longrightarrow \dot{q}^{(i)}(t) + \delta \dot{q}^{(i)}(t)$$

with $i = 1, \ldots, N$. The action of the system is given by

$$S(q,\dot{q}) = \int L(q,\dot{q})dt$$

where q and $\dot{q}$ are n-dimensional column vectors. The action differential

$$\begin{aligned}
\delta S &= \int L(q+\delta q, \dot{q}+\delta\dot{q})dt - \int L(q,\dot{q})dt \\
&= \int L(q+\delta q, \dot{q}+\delta\dot{q})dt - \int L(q,\dot{q})dt \\
&= \int \left[\sum_i \frac{\partial L}{\partial q^{(i)}}\delta q^{(i)} + \sum_i \frac{\partial L}{\partial \dot{q}^{(i)}}\delta \dot{q}^{(i)}\right] dt \\
&= -\int \sum_i \left[\frac{d}{dt}\frac{\partial L}{\partial \dot{q}^{(i)}} - \frac{\partial L}{\partial q^{(i)}}\right]\delta q^{(i)} dt \\
&= -\sum dt \sum_i E_i^{(o)}(q,\dot{q},\ddot{q})\,\delta q^{(i)}
\end{aligned}$$

where we define the Euler-Lagrange differential operator

$$E_i^{(o)}(q,\dot{q},\ddot{q}) = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}^{(i)}} - \frac{\partial L}{\partial q^{(i)}}.$$

Note that

$$\int \sum_{i=1}^{N} E_i^{(o)}(q,\dot{q},\ddot{q})\delta q^{(i)} dt \equiv 0 \tag{6.17-1}$$

on shell. Expanding $E_i^{(o)}$

$$\begin{aligned}
E_i^{(o)}(q,\dot{q},\ddot{q}) &= \sum_j \left[\frac{\partial^2 L(q,\dot{q})}{\partial \dot{q}^{(i)}\partial \ddot{q}^{(j)}}\ddot{q}^{(j)} + \frac{\partial^2 L(q,\dot{q})}{\partial \dot{q}^{(i)}\partial q^{(j)}}\dot{q}^{(i)}\right] - \frac{\partial L(q,\dot{q})}{\partial q^{(i)}} \\
&= \sum_j W_{ij}(q,\dot{q})\ddot{q}^{(j)} + \sum_j \frac{\partial^2 L(q,\dot{q})}{\partial \dot{q}^{(i)}\partial q^{(j)}}\dot{q}^{(i)} - \frac{\partial L(q,\dot{q})}{\partial q^{(i)}}
\end{aligned}$$

$$= \sum_j W_{ij}(q,\dot{q})\ddot{q}^{(j)} + k_i(q,\dot{q})$$

If L is singular, $W_{(N\times N)}$ is not invertible so (6.17-1) cannot be solved for $\ddot{q}_i, i = 1, \dots, N$. If $\mathrm{Rank}\big(W(q,\dot{q})\big) = R_W$ on shell, then there exist $N - R_W$ in the theory. There exist $N - R_W$ independent left (or right) zero mode eigenvectors $w_i^{(o,k)}, i = 1, \dots, N - R_W$ such that

$$\sum_i w_i^{(o,k)}(q,\dot{q})W_{ij}(q,\dot{q}) = 0, \qquad k = 1, \dots, N - R_W$$

(6.17-2)

Thus

$$\phi^{(o,k)} = \sum_{i=1}^{N} w_i^{(o,k)}(q,\dot{q})E_i^{(o)}(q,\dot{q},\ddot{q})$$

depend on q and $\dot{q}$ only. The $\phi^{(o,k)}$ also vanish on shell:

$$\phi^{(o,k)}(q,\dot{q}) = 0, \quad k = 1, \dots, N - R_W$$

The set $\{\phi^{(o,k)} | k = 1, \dots, N - R_W\}$ are the zero generation constraints. It is possible that *not* all the $\{\phi^{(o,k)}\}$ are linearly independent. So we may find linear combinations of the zero mode eigenvectors

$$v_i^{(o,n_o)} = \sum_k c_k^{(n_o)} w_i^{(o,k)}$$

such that we have

$$G^{(o,n_o)} = v^{(o,n_o)}E^{(o)} \equiv 0, \qquad n_o, \dots, N_o$$

(6.17-3)

These are called gauge identities.

Any variation δq_i, $i = 1, \dots, N$, of the form

$$\delta q_i = \sum_{n_o} \varepsilon_{n_o} v_i^{(o,n_o)}$$

Is action invariant by (6.17-1). Given this definition of δq_i and (6.17-3), we conclude

$$\begin{aligned}\delta S &= \int dt \sum_{i=1}^{N} E_i^{(o)}(q,\dot q,\ddot q) \sum_{n_o} \varepsilon_{n_o}(t) v_i^{(o,n_o)} \\ &= \int dt \sum_{i=1}^{N} \varepsilon_{n_o} \sum_{n_o} E_i^{(o)}(q,\dot q,\ddot q) v_i^{(o,n_o)}(q,\dot q) \\ &= \int dt \sum_{i=1}^{N} \varepsilon_{n_o} G^{(o,n_o)} \\ &\equiv 0\end{aligned}$$

everywhere. The remaining zero generating modes which we denote by $u^{(o,n_o)}$ lead to genuine constraints. They are of the form $\phi^{(o,n_o)}(q,\dot q) = 0$ on shell, where

$$\phi^{(o,n_o)} = u^{(o,n_o)} E^{(o)}.$$

(6.17-4)

The algorithm now proceeds as follows. We separate the gauge identities (6.17-3) from the nontrivial constraints (6.17-4) and will list them separately. They will be used for determining *local symmetry transformations*.

Next we want to search for additional constraints. We do this by searching for further functions of the coordinates and velocities which vanish in the space of physical trajectories. To this effect consider the following $N + N_o$ vector constructed from $E^{(o)}$ and the time derivative of the constraints (6.17-4)

$$[E^{(1)}] = \begin{bmatrix} E^{(o)} \\ \frac{d}{dt}\left(u^{(o,1)} E^{(o)}\right) \\ \vdots \\ \frac{d}{dt}\left(u^{(o,n_o)} E^{(o)}\right) \end{bmatrix} = \begin{bmatrix} E^{(o)} \\ \frac{d}{dt}\phi^{(o)} \end{bmatrix}$$

(6.17-5)

by construction. The constraint $\phi^{(o)}$ is valid for all time and therefore $\frac{d}{dt}\phi^{(o)} = 0$ on shell, but

$$\frac{d\phi^{(o,i)}}{dt} = \nabla \dot q\left(u^{(o,i)} E^{(o)}\right)\ddot q + \nabla q\left(u^{(o,i)} E^{(o)}\right)\dot q$$

(6.17-6)

So

$$\left[E_{i_1}^{(1)}\right] = \sum_{j=1}^{n} W_{i_1 j}^{(1)}(q,\dot{q})\dot{q}^{(j)} + k_{i_1}^{(1)}(q,\dot{q})$$

where $i_1 = 1, \ldots, N + N_o$, and

$$\left[W_{i_1 i}^{(1)}\right] = \begin{bmatrix} W^{(o)} \\ \nabla\dot{q}\left(u^{(o,i)}E^{(o)}\right) \\ \vdots \\ \nabla\dot{q}\left(u^{(o,N_o)}E^{(o)}\right) \end{bmatrix}$$

$$\left[k_{i_1}^{(1)}\right] = \begin{bmatrix} k^{(o)} \\ \sum_j \frac{\partial}{\partial q^{(j)}}\left(u^{(o,i)}E^{(o)}\right)\dot{q}^{(j)} \\ \vdots \\ \sum_j \frac{\partial}{\partial q^{(j)}}\left(u^{(o,N_o)}E^{(o)}\right)\dot{q}^{(j)} \end{bmatrix}$$

(6.17-7)

We next look for the *zero modes* of $W^{(1)}$. By construction, these zero modes include the *o* modes of the previous level. The gauge identities at level 1 are.

$$G^{(1,n_1)} = v^{(1,n_1)}E^1 - \sum_{n_o=1}^{N_o} M_{n_1 n_o}^{(1,0)}\left(u^{(o,n_o)}E^{(o)}\right) \equiv 0$$

(6.17-8)

where $n_1 = 1, \ldots, N_1$ and the genuine constraints are of the form

$$\phi^{(1,n_1)} = \phi^{(1,n_1)}E^1 = 0$$

(6.17-9)

with $n_1 = 1, \ldots, N_1$ on shell.

We next adjoin the new identities (6.17-8) to the ones determined earlier (6.17-3) with the remaining constraints (6.17-9) we proceed as before, adjoining their time derivatives to (6.17-5) and construct $W_{i_1 i}^{(1)}$ and $k_{i_1}^{(1)}$.

The iterative process will terminate at some level M if either i) *there is not further zero modes*, or ii) *the new constraints can be expressed as linear combinations of previous constraints*.

6.17.1 The maximal set of linearly independent gauge identities generated by the algorithm

Note that the algorithm steps are of the form

$$G^{(o,n_o)} = u^{(o,n_o)} E^{(o)} \equiv 0$$

(6.17-10)

$$G^{(l,n_l)} = u^{(l,n_l)} E^{(l)} - \sum_{l'=0}^{l-1} \sum_{n_{l'}=0}^{N_{l'}} M_{n_l n_{l'}}^{(l,l')} \phi^{(l',n_{l'})}$$

(6.17-11)

with $L = 1, \dots, N_l$. The $M_{n_l n_{l'}}^{(l,l')}$ are only functions of q and $\dot{q}$. And

$$\phi^{(l,n_l)} = u^{(l,n_l)} E^{(l)}, \quad n_l = 1, \dots, N_l,$$

(6.17-12)

$$E^{(l)} = \begin{bmatrix} E^{(o)} \\ \dfrac{d\phi^{(o)}}{dt} \\ \vdots \\ \dfrac{d\phi^{(l-1)}}{dt} \end{bmatrix}$$

(6.17-13)

where $\phi^{(l)}$ is a column vector with N_l components $\phi^{(l,n_l)}$. Thus we conclude from (6.17-13) and (6.17-11) that the general form of the gauge identity given by (6.17-11) is of the form

$$G^{(l,n_l)} = \sum_{i=1}^{N_l} \sum_{l=1}^{M} \sum_{m=1}^{l} \varsigma_{mi}^{(l,m_l)} \frac{d^m}{dt^m} E_i^{(o)} \equiv 0$$

(6.17-14)

where $\varsigma_{mi}^{(l,m_l)}(q, \dot{q})$ and $N_l < M$. From (6.17-14) it also follows that

$$\sum_{l=1}^{M} \sum_{n_l=1}^{l} \varepsilon^{(l,n_l)} G^{(l,n_l)} \equiv 0$$

(6.17-15)

This identity can also be written as

$$\sum \delta q^{(i)} E_i^{(o)} - \frac{d}{dt} F$$

where

$$\delta q^{(i)} = \sum_{l=1}^{M} \sum_{n_l=1}^{N_l} \sum_{m=q}^{l} (-1)^m \frac{d^m}{dt^m} \varsigma_m^{(l,n_l)} \varepsilon^{(l,m_l)}(t)$$

(6.17-16)

and F is a complicated function of q and $\dot{q}$. By collecting indices l, n_l together

$$\delta q_i = \sum_{l=1}^{M} \sum_{n_l=1}^{N_l} \sum_{m=q}^{l} (-1)^m \varsigma_{m_i}^{(a)} \varepsilon^{(a)}(t)$$

6.17.2 Example of constrained Hamiltonian system in Lagrangian form

Let

$$L(q,\dot{q}) = \frac{1}{2}\dot{q}^{2(1)} + \dot{q}^{(1)} q^{(2)} + \frac{1}{2}\left(q^{(1)} - q^{(2)}\right)^2$$

(6.17-17)

$$E^{(o)} = \begin{bmatrix} \frac{d}{dt}\frac{\partial}{\partial \dot{q}^{(1)}} - \frac{\partial L}{\partial q^{(1)}} \\ \frac{d}{dt}\frac{\partial}{\partial \dot{q}^{(2)}} - \frac{\partial L}{\partial q^{(2)}} \end{bmatrix} = \begin{bmatrix} \ddot{q}^{(1)} + 2q^{(2)} - q^{(1)} \\ q^{(1)} - q^{(2)} \end{bmatrix}$$

(6.17-18)

$$W = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

(6.17-19)

$$k = \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \end{bmatrix}$$

(6.17-20)

The only o mode is

$$u^{(o)} = [0,1]$$

Then

$$\begin{aligned} E^{(o)} &= W^{(o)}\ddot{q} + k^{(o)} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q}^{(1)} \\ \ddot{q}^{(2)} \end{bmatrix} + \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \end{bmatrix} \end{aligned}$$

Then

$$\begin{aligned} u^{(o)}E^{(o)} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \left[\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q}^{(1)} \\ \ddot{q}^{(2)} \end{bmatrix} + \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \end{bmatrix} \right] \\ &= -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \\ &= 0 \end{aligned}$$

(6.17-21)

on shell. Then there are no gauge identities for $E^{(o)}$. Now construct $E^{(1)}$.

$$E^{(1)} = \begin{bmatrix} E^{(o)} \\ \frac{t}{dt} u^{(o)}E^{(o)} \end{bmatrix} = \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \\ -\ddot{q}^{(1)} - \dot{q}^{(2)} + \dot{q}^{(1)} \end{bmatrix}$$

which can be written

$$\begin{aligned} E^{(1)} &= W^{(1)}\ddot{q} + k^{(1)} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \ddot{q}^{(1)} \\ \ddot{q}^{(2)} \end{bmatrix} + \begin{bmatrix} \dot{q}^{(2)} - q^{(1)} + q^{(2)} \\ -\dot{q}^{(1)} - q^{(2)} + q^{(1)} \\ -\dot{q}^{(2)} + \dot{q}^{(1)} \end{bmatrix} \end{aligned}$$

There zero modes of $W^{(1)}$ are

$$W^{(1)} \begin{cases} \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \\ \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \end{cases}$$

The first zero mode is the previous one augmented by one dimension and reproduces the previous constraint. The second mode reproduces the negative of the constraint (6.17-21). That is,

$$v^{(1)}E^{(1)} = -u^{(o)}E^{(o)}$$

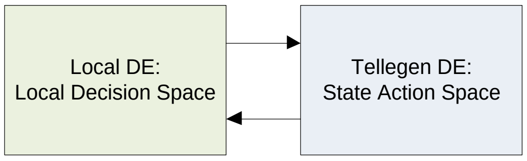
with $v^{(1)} = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}$. This leads to the gauge identity

$$G^{(1)} = v^{(1)}E^{(1)} + u^{(o)}E^{(o)} \equiv 0$$

6.18 Companionship: Reconciling agents in the network.

The outline of the companionship process is as follows for a system of N agents.

1. Determine the *state action space* of the system for $N - 1$ agents to create a *Tellegen decision element*.
2. Update the remaining agent with the Tellegen DE so that the ?? is minimized.
3. Repeat process so that all N agents are updated with respect to their Tellegen DEs.



7 Limitations

7.1 Response time

TBD

8 Architectural flow

This section show diagrams of various key aspects of the architecture.

8.1 Deployment Architecture

In a separate document

8.1.1 General description of architecture

In a separate document

8.1.1.1 Points of scalability

Points of scalability are the architectural aspects of the system that are required to scale. These include:

- The number of DE's that can contribute to query resolution.
- The number of variables that can contribute the query resolution.
- The number of rules that can contribute to query resolution.

8.1.1.2 Single points of failure

TBD

8.2 High-level Flows

8.2.1 The Distributed Architecture (DA)

Show flows for the main things that happen in the DA.:

8.2.1.1 Flow for query resolution.

The following diagram provides an overview of query resolution for a particular decision element.

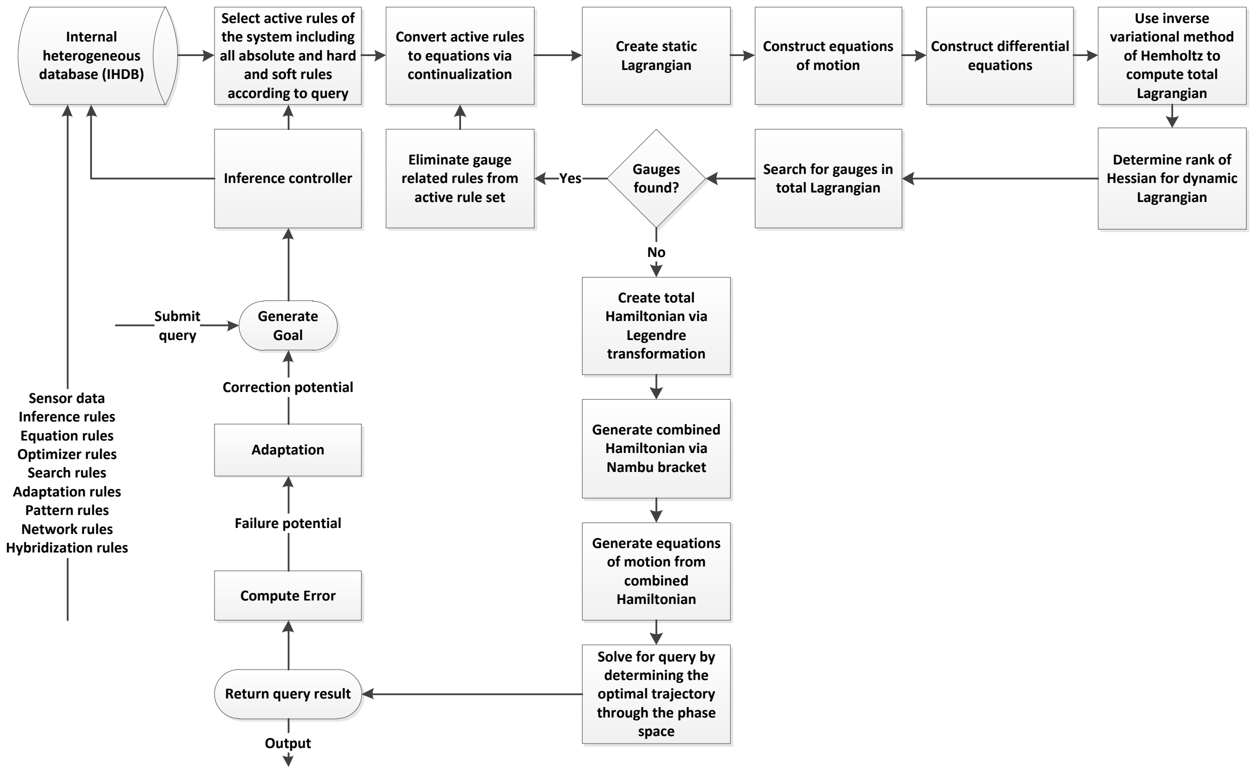
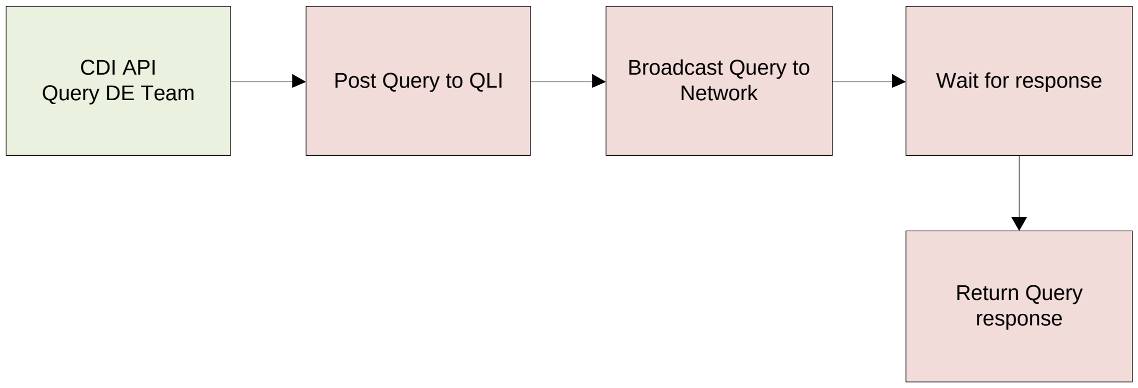
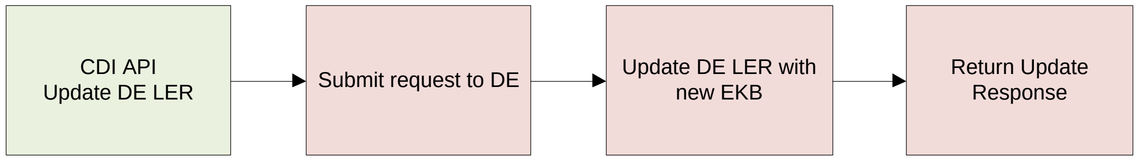

Internal heterogeneous database (IHDB)
Select active rules of the system including all absolute and hard and soft rules according to query
Convert active rules to equations via continualization
Create static Lagrangian
Construct equations of motion
Construct differential equations
Use inverse variational method of Hemholtz to compute total Lagrangian
Inference controller
Eliminate gauge related rules from active rule set
Yes
Gauges found?
Search for gauges in total Lagrangian
Determine rank of Hessian for dynamic Lagrangian
No
Submit query
Generate Goal
Create total Hamiltonian via Legendre transformation
Correction potential
Sensor data
Inference rules
Equation rules
Optimizer rules
Search rules
Adaptation rules
Pattern rules
Network rules
Hybridization rules
Adaptation
Generate combined Hamiltonian via Nambu bracket
Failure potential
Generate equations of motion from combined Hamiltonian
Compute Error
Return query result
Solve for query by determining the optimal trajectory through the phase space
Output

Fig. 8.2-1

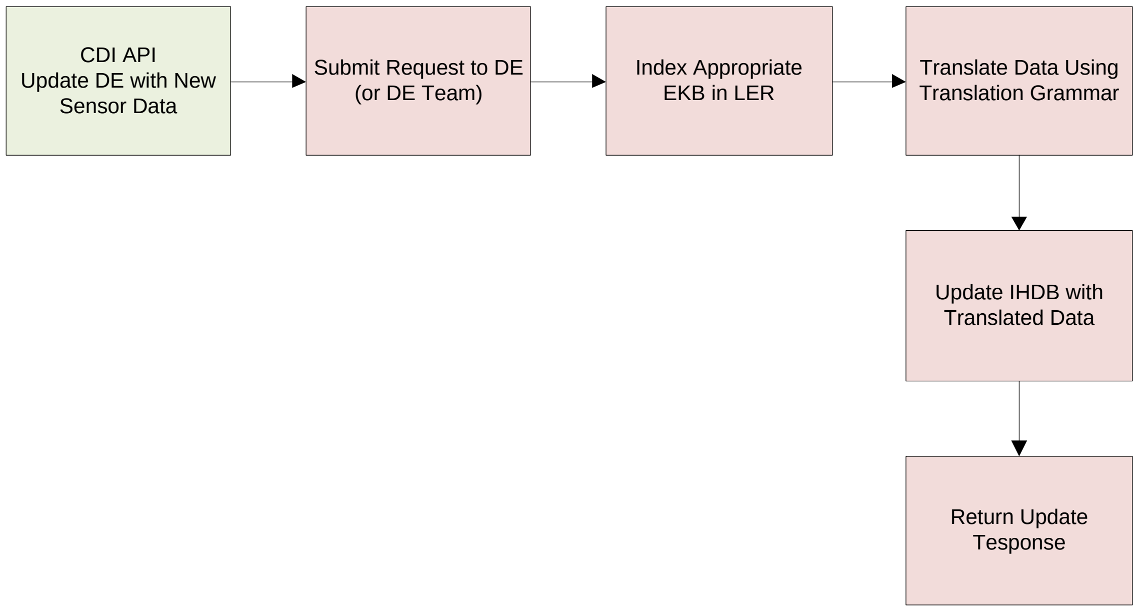
- User submits query.
- System used KB to establish equations of motion for system in Lagrangian or Hamiltonian form.
- System determines optimal trajectory via optimization algorithm of the equations of motion that conform to the principle of least action.
- System returns solution which is a point in the phase space and also serves as an answer to the query.



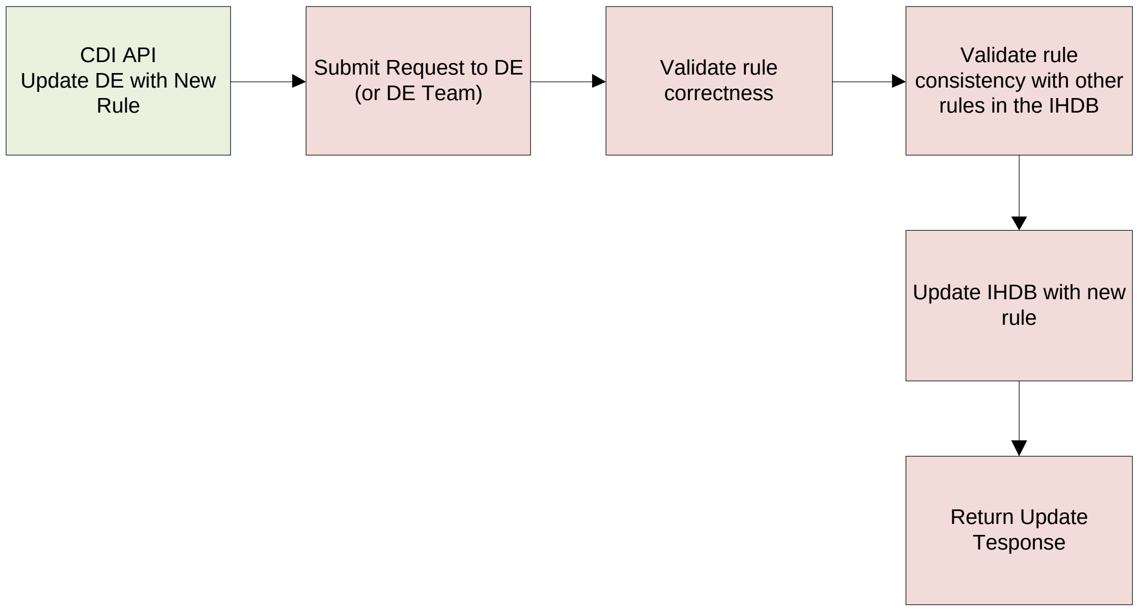
8.2.1.2 Flow for updating DE's with new external repositories.



8.2.1.3 Flow for updating DE's with new sensor data.


CDI API
Update DE with New Sensor Data
Submit Request to DE (or DE Team)
Index Appropriate EKB in LER
Translate Data Using Translation Grammar
Update IHDB with Translated Data
Return Update Tesponse

8.2.1.4 *Flow for updating DE's with new rules.*


CDI API
Update DE with New Rule
Submit Request to DE (or DE Team)
Validate rule correctness
Validate rule consistency with other rules in the IHDB
Update IHDB with new rule
Return Update Tesponse

8.2.2 The Internal Heterogeneous Database (IHDB)

Show flows for the main things that happen with the IHDB:

- Flow for adding new rule to the IHDB.
- Flow for updating an existing rule to the IHDB.
- Flow for deleting rule from the IHDB.
- Flow for consistency check for the IHDB.

8.2.3 The Rule Entry Interface (REI)

Show flows for the main things that happen with the REI:

- Flow for submitting rule to the REI. This should include a validity check and error handling.

8.2.4 The Rule Editor (RE)

Show flows for the main things that happen with the RE:

- Flow for creating rule within the RE including variable selection, syntax checking, and test evaluation.

8.2.5 The External Knowledge Base (EKB)

Show flows that happen with the EKB:

- Flow for updating the EKB schema.
- Flow for updating the EKB with new sensor data. This may include some synchronization if there are to be multiple EKB's.
- Flow for notifying DE's that new sensor data is available.

8.2.6 The Sensor Ingestion Interface (SII)

Show flows that happen with the SII:

- Flow for adding a new sensor to the network.
- Flow for polling a sensor.
- Flow for submitting data to the network.
- Flow for deleting a sensor from the network.

8.2.7 The Rule Conversion Engine (RCE)

Show flows that happen with the RCE:

- Flow for identifying rule sets.
- Flow for compiling rule sets into equational form.

8.2.8 The Decision Element (DE)

Show flows that happen with the DE:

- Flow for updating the LER.
- Flow for submitting a query to the DE.
- Flow for returning response to a query from the DE.
- Flow for operating the programmable search engine.

8.2.9 The Query Language Interface (QLI)

Show flows that happen with the DE:

- Flow for formulating a query and submitting to the DA.
- Flow for receiving a response to a query from the DA.

8.2.10 The Minimization Function Generator (MFG)

Show flows that happen with the MFG:

- Flow for formulating translating a query to a minimization function.

8.2.11 The Query Response Engine (QRE)

Show flows that happen with the QRE:

- Flow for finding minimum of the minimization function.

8.2.12 The Pareto Multi-Criteria Optimization Engine (PMOE)

Show flows that happen with the QRE:

- Flow for companionship.

8.3 Complex Use Cases

These are flows for use cases that utilize the various aspects of the architecture:

- Commitment planning
- Nonlinear feedback system
- Distributed sensor network with local and aggregate prediction for weather based attributes

9 Software realization of the architecture

This section describes key software packages and their corresponding roles in the architecture. An example would be to define Lucene and discuss its role in indexing and search. The purpose is to provide clarity around what software we are using and what it does to support the architecture.

9.1 Production

9.1.1 Language requirements

9.1.1.1 Java

All production code will typically use Java for implementation. Cases when other languages are used are when existing libraries are required for operation based on other languages.

9.1.2 Persistence requirements

9.1.3 Messaging and request queuing requirements

9.1.4 Mathematical algorithm requirements

9.1.5 API requirements

9.1.6 Agent related requirements

9.2 Tools

9.2.1 Eclipse IDE for Java

Eclipse is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system. [Wikipedia]

Eclipse is the default Java development environment for Veritone.

9.2.2 Prolog

Prolog is used for system prototyping.

9.2.3 Python

In a separate document

10 Data exchange protocols

The system will have needs around the exchange of data in and out of the system and among the various subcomponents. For example, SOAP defines an interface for web services. Thrift and Avro are lighter weight data exchange protocols that enable more rapid development cycles and may be appropriate for subcomponents that are tightly coupled.

What is the default case?

The default communication protocol for our components should be lightweight, fast, easy to debug and easy to develop; a good choice is to develop component APIs using Jersey and hosting them in Tomcat.

10.1 Veritone API

REST HTTP (GET, POST, PUT, DELETE)

11 Environment

11.1 Development Environment

Defines the development configuration environment. (Needs to be updated.)

Component	Version	Notes
Linux		
Python	1.6.0_23 (i.e. Java 6 Update 23)	

11.2 Deployment Environment

Defines the deployment configuration environment.

Component	Version	Notes
Linux	7	
Python	1.6.0_23 (i.e. Java 6 Update 23)	

11.3 Infrastructure Design

This section discusses the infrastructure design which includes the following.

- Hardware components that define a deployment.
- Types of servers and the expected provisioning for a server.
- Deployment configuration given the server types.
- Startup processes for servers.
- Sample diagram for active deployment.
- Cost analysis of various deployment configurations given environments.
- Sensor networks.

12 Data

The software is designed to handle certain types of data. It is important to highlight key requirements and limitations around the type of data the system is expected to process. Additionally, the system will produce data for users of the system. This is also defined. Frequently, there will be a standard data specification document which discusses system requirements and expectations around data. This is summarized here.

12.1 Inputs

Inputs to the system are provided administratively through use of the API and by users who submit queries and add preferences to their profiles.