Intelligent Self Reliance Agent (ISRA) Protocol

The Full Articulation of Functional Artificial General Intelligence Prototype

What is inside

-The goal of the project itself and expectations.
-The implementation documentation of the protocol.
-The variations that could be applied and their consequences.

Goal

Provide a platform for everyone to share, contribute to global collective knowledge base, make use of it to automate repetitive cognitive tasks and solve world problems.

Disclaimer

Some idea that exist on this paper may / may not been introduced elsewhere but didn't included in the reference section due to:
-Incident (Not knowing)
-Independently introduced
-General knowledge (Common sense)
Author have by no mean to plagiarize anything, thus if found anything that belongs to established theory and paper but not cited, do contact us.

It is designed to be read top down, jumping around pages is not advised unless seeking for word definition.

The term “we” and “us” refers to the development team.
“our” means we, us. “he”, “him”, “it” refers to the protocol.

The document is provided 'AS IS' without warranty of any kind, in no event shall the author be liable for any claim, damages or other liability resulting from or related to the use of this document.
Reserved the right to change any text in the future without further notice.
Free to distribute or modify.

Author

Ong Guan Xing
Email: xing12397@gmail.com
Or visit www.ogxis.com for latest link in case email broken.

Table of Contents

General flow of the system (non technical)

From external to internal:
Data[1]comes in from the outside world.
Identify what those data are, are they relevant to what the system is doing now.
If relevant then maintain course, else think whether or not to change course.

Internally always have a focus on the things that the system want to do (course).
The select criteria is:
It should be unique most of the time[2].
It is relevant to current situation[3].
It is more important than all other views[4].

Based on the selected attention point, generate solution based on experience in order to recreate it.
This will boost new experience generation and increase confidence for the particular actions.

Generate a solution tree[5] based on experience, then from it select the best path to execute.
At this moment the path is not executed yet, it is on his mind and ready to execute, but the time hasn't arrived yet.

So it can be viewed as prediction, he arrange and predict what will happen at the near future.
Then he forward those data into working memory[6] for scheduling and executing purposes.
That free up his mind to do other computations[7].

If all the predictions goes well, he increase confidence on that particular subject.
If it fails, he will be notified and forced to switch to another solution from the solution tree he just generated, if it contains this unexpected situation, it means it is expected and he can just switch to that branch and execute, else he will have to generate a new tree as the no solution is available to solve that unexpected situation.

These processes will go on forever iteratively and it will create a mind system that is tolerant to unexpected situation with quick thinking to resolve it and get back on track to finish its task, able to accustom to any places as it doesn't requires any prerequisite to learn, experience are cross collate able thus can be reused on different occasions and as it is built up from scratch and pile up on experience.

With default inclination toward uniqueness (curiosity), it enable the system to be on constant learning phrase and promote creativity.

Together it allows ultimate generalization, abstract thinking, creativity and traits formulation to take place.

---

1 Data can be of any type of sensor's input.
2 It can be unique or non unique based on current criteria by valuating *polyVal27*.
3 Relevance are calculated by comparing *polyVal* against *globalDist27*.
4 Valuated via importance of the course's implication.
5 *Solution tree* is a tree structure containing potential next step that the protocol could take to attain its current goal.
6 *Working Memory* is similar to people's biological counterpart.
7 Mind are composed of *Working Memory* (Short-term memory), *Databases* (Long term memory) and computational threads (Data Manipulators, all the functional processes that carries out cognitive tasks).
* All the *italicized* entity will be explained in detail individually in upcoming text.

Implementation overview

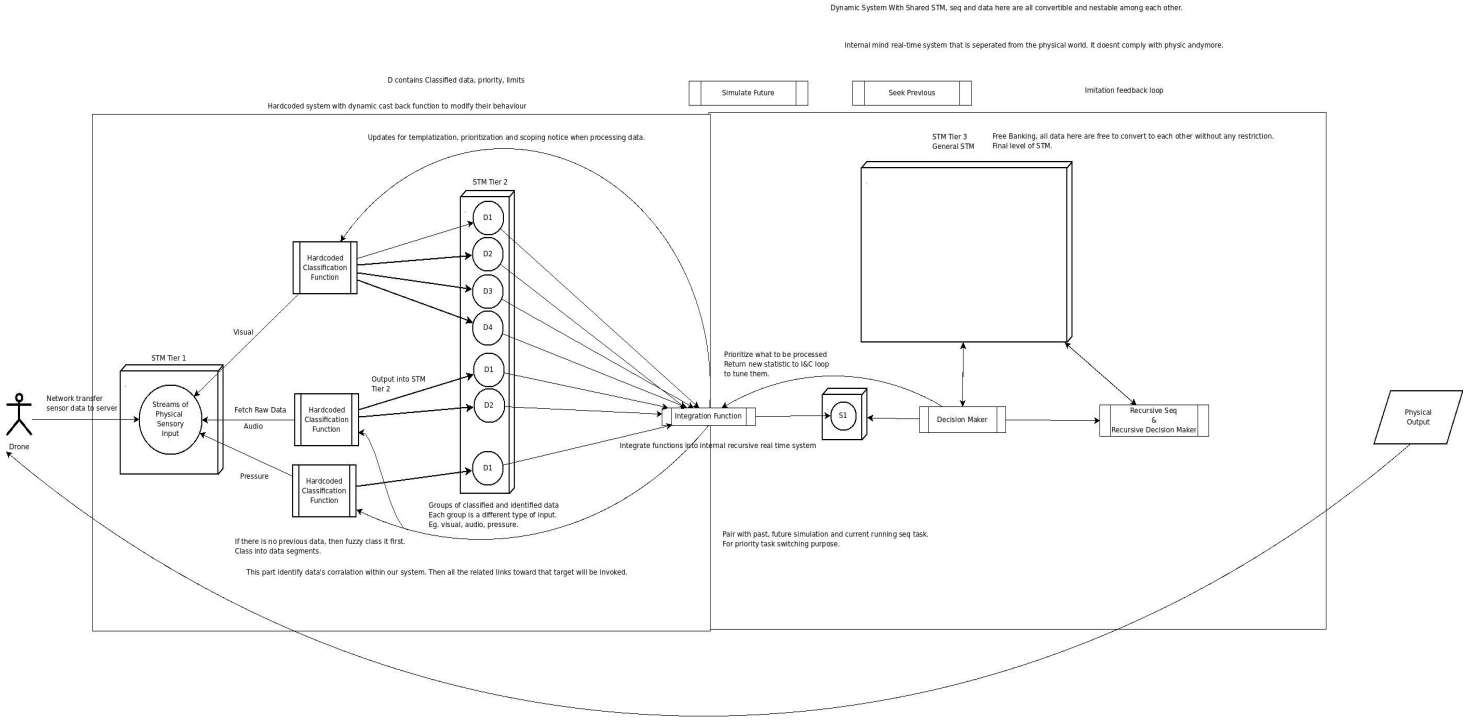


Diagram: The visualized flow of the overall operations.

External Layer

Sensor Layer

-Sensor that receive data from real world and make the data accessible by internal mind.
-Sensor that send current position as feedback to internal mind and execute commands sent by it.
Sensor can be any sensor.
The default sensor group are visual, audio and all other sensors in analog form.

Internal Layer

Pattern Processing layer

-Internal system run 'Identification and Classification loop' (ICL) to process raw sensor input (including feedbacks) into segment of recognized patterns or disintegrate them into logical pattern chunks if they are not recognizable.
-Calculate the mean of the just recognized OR generated pattern.
-Group them together into a logical unit using a process called Global Changes Association (GCA).

Working Memory Layer

-Constantly get latest update into its working set of data.
-Select new attention point.
-Prediction against Reality check (PaRc).
-Contagious memory generation in experience format after each successful mind operation.

General Processing Layer

-Generate solution tree based on experience for the selected attention by working memory.
-If PaRC fails, switch to the next best solution from the solution tree initiated by working memory via message or flag, if no further solution available, generate a new tree with the same goal.
-Calculate and set Global Distribution value.
-Post predictions to WM.

Sensor and Pattern Processing layer

Flow

Sensor data are directly ported/made available for fetching to pattern processing layer for processing using ICL.

ICL will process all incoming raw sensory input and select whether to use specialized ICL depending on their type.

Visual and audio data are encoded in such a special format that it cannot be processed using the default ICL implementation designed for general analog based sensor input without significant conversion.

Therefore it must be provided with specialized ICL routines in order to extract and identify patterns from them separately, those algorithm are not required to be perfect, any open source recognition library can do the deal.

Rationale

Any data in existence can be has the properties of wave, and thus can be described in waveform according to quantum mechanics[8].

No matter how complex it seems, a wave can be decomposed into multiple (or possibly infinite) simple sine waves according to fourier series[9].

Therefore for the universal solution it must possess the ability to ICL these basic waveform.
As the decomposed sine waves are always analog, it is set as default.

Visual and audio signals although in theory can be decomposed into simple sine waves and be processed by the default ICL, doesn't do that due to:
-Complexity involved (decomposition processes not well defined nor understood)[10]
-Loss of meaning (data has to be decoded to bring out multitude of original meanings and intents)[11]

Thus specialized decoding and ICL scheme will be defined for them independently.

8 Quantum mechanics video: https://www.youtube.com/watch?v=Io-HXZTepH4 by Eugene Khutoryansky.
9 Fourier series video: https://www.youtube.com/watch?v=r18Gi8lSkfM by Eugene Khutoryansky.
Wiki: https://en.wikipedia.org/wiki/Fourier_series
10 How to separate an image into multiple waves that can at the same time represent particular spectrum while at the same time exhibits its coordinate and compress these data into a single sine wave?
How and how not to separate audio signals so that the outputted signals remain significant/meaningful?
11 Encoding scheme are designed to represent multiple aspect of the data in a binary stream, default ICL expects pure analog signals, thus decoding has to be done before feeding into it and figure out whether or not to separate those aspects to feed to one or more ICL.

Default ICL (Sensor ICL)

Default is aimed at the most fundamental type of sensory input, pure analog signals with no additional encoding scheme alongside with boundary range (max min upper lower bound).

Requirements

-Must be analog signals (does not contain complex object type, only primitive type allowed).
-Altered to represent themselves in form of range 0~100 double.
-Encapsulate the translation between 0~100 range (polyVal) to actual sensor output range (may be any range depending on the sensor).

Implementation

Initialize all new type of analog sensor input using provided data boundary.

Clip them into polyVal range (0~100 double) alongside with a transition mapping function that translate equivalent analog data to equivalent polyVal representation.

Begin to accept new raw data and for each of them:
No identification is required as they are pure analog data[12].
Treat those raw data as patterns.
Forward those pattern to Working Memory via GCA to acknowledge the whole system about the existence of these newly arrived pattern data.

Application

Any analog sensors, can be read only, writable or both.
Some examples:
Read only:
-Temperature sensor
-Hall effect sensor
-Pressure sensor
-Barometer
-Proximity sensor

Writable:
-Motors
-LEDs
-Relays
And any more analog based IO component.

12 As these are analog signals, no pattern is extractable from the signal as it is already in its pure form, thus it is valid to consider it itself as the final pattern thus no more extraction is required (treat it as a whole). For example stepper motor input, which is just an analog signal telling us its current position.

Visual ICL

Requirements

-Produce the same output if fed with the same data given there is no external influence. (functional)
-Able to match pattern in scale invariant way.
-Able to match pattern extracted from the same frame.
-A controllable crude pattern/feature extraction algorithm.
-All input regardless of where it comes from (MPEG streams or individual images or anything else), must provide a way decode it into individual images storable inside a raw matrix.

Implementation

No initialization is required as the boundary of image data is dynamically deducible via the image encoding by reading the size of the matrix containing the decoded data.

Clipping should already been done in code as image data are always going to be 0~255 lower upper boundary, thus the transition mapping function can be hard coded.

Begin to accept new raw image data and for each of them:
If Working Memory has provided us with expected patterns, match those patterns against raw image to see if they exist.
If exist, mark that region as recognized and record this pattern as an occurrence of the given pattern.
Else notify Working Memory that the specified pattern doesn't match anything (PaRc fail[13]).

After depleting the given pattern or no pattern is given at all,
For all remaining unrecognized (unprocessed) image area, run a find contour operation.

For each contour found that are within unrecognized area, treat them as a new pattern.

Forward those pattern to Working Memory via GCA to acknowledge the whole system about the existence of these newly generated pattern data.

All these algorithm (pattern extraction, template matching scale invariant[14], contour finding, contour extraction and image decoder into matrix of 8 unsigned char 3 channel (8UC3) are all available in the opencv library.

Application

Any camera that uses any type of encoding can be used as long as it can be decoded to fill in an image matrix.

13 Prediction against Reality check failure (PaRc12), a Working Memory operation.
14 Scale invariant means the pattern image size can be bigger or smaller (scaled) but still being able to be recognized. Rotation preferred to be treated as not match, therefore orientation is important.

Audio ICL

Requirements

-Produce the same output if fed with the same data given there is no external influence. (functional)
-Rudimentary pattern extraction based on any peak detection algorithm as long as the outputted pattern contains some form of isolation betweens data (eg. Silent point).
-Able to match pattern extracted from the same sample.
-All input regardless of where it comes from (WAVE streams or individual audio segment or anything else), must provide a way decode it into an audio waveform.

Implementation

No initialization is required as the boundary of audio data is dynamically deducible via the encoded audio header or directly counting the size of the data byte array.

Clipping should already been done in code as audio data are always going to be -1 ~ 1 float lower upper boundary, thus the transition mapping function can be hard coded.

Begin to accept new raw audio data and for each of them:
If Working Memory has provided us with expected patterns, match those patterns against raw audio data to see if they exist.
If exist, mark that region as recognized and record this pattern as an occurrence of the given pattern.
Else notify Working Memory that the specified pattern doesn't match anything (PaRc fail[15]).

After depleting the given pattern or no pattern is given at all,
For all remaining unrecognized (unprocessed) image area, run a general rudimentary pattern extraction algorithm which detects silent point or where peaks begin and end sequences in order to extract crudely meaningful pattern from the audio stream[16].

For each pattern extracted that are within unrecognized area, treat them as a new pattern.

Forward those pattern to Working Memory via GCA to acknowledge the whole system about the existence of these newly generated pattern data.

No recommendation on any algorithm or library to use as there is no de facto industrial standard library when it comes to audio processing.

Application

Any microphone or recording devices that uses any type of encoding can be used as long as it can be decoded into an audio stream.

15 Prediction against Reality check failure (PaRc12), a Working Memory operation.
16 For simplicity, a peak is considered more meaningful than silence or constant background noise by common sense. Thus it qualifies as a breakpoint between interesting and not so meaningful data.

Working Memory Layer

Rationale

Inspired by brain's working memory, with similar functionality.
A place for internal worker threads to share and exchange data system wide.
A place for experience to converge and form by cross relating with other experiences.
A place to create and refine linkage between experiences.
A scope of current interest and at the moment awareness.
A place to form collective swarm intelligence implicitly by synchronizing the whole system both in and outside the big picture automatically.

Algorithms

Each of them work independently but together create a global ecosystem for intelligence to florish.
-Select new attention point.
-Maintain list of currently addressable vertexes based on timing and relevancy, mainly direct import from latest GCA and manage their lifecycle. If they are no longer needed or expired it will be disposed from the WM, but the actual record itself still exist (Short term memory, STM).
-Prediction against Reality check (PaRc) alongside with a scheduling system in order to compute time based solutions.

Fetch from GCA

WM will constantly update its current working vertex set by fetching from GCA engine to keep himself in sync with the current global situation.
All new raw sensor data will only be visible globally after they enter here.

Select new attention point

Every decision tree[17] begins here. WM will select the most appropriate vertex from the whole working memory set imported using core decision logics[18].
Then forward that selected vertex to the main computing system[19] to generate solutions and decision trees (plan next operation).
One instance of this worker can only fetch 1 exp vertex at any time as its focus.
As brain can only effectively deal with 1 thing at a time, mixing multiple task together creates confusion and reduces efficiency.
Rapid attention point switching however, is allowed[20].

17 Decision tree is a tree full of possible solutions to accomplish the given goal. The goal here is the recreate the newly selected attention point. Its structure will be discussed in detail later.16
18 Core decision logics are just sets of simple constrains, will be discussed in detail later.22
19 Main computing system are series of operations that cover decision tree generation, which includes experience fetching, validating, selection, rearrangement according to time, composite experience evaluation to seek of incontinence before adding it as solution.
20 Feasible without having to do anything, if the system sense it is required, he will arrange them in advance by embedding the switches within the decision tree, there is no random switching, but are all prepared in advance.

Prediction against Reality check (PaRc)

The main computing system will return us with the decision tree and what path he had selected from it. Then WM will execute and keep track of the execution progress, keeping an eye on for any abnormality.

Rationale

WM is the only place latest data keep coming in and out.
And with prediction (decided route[21], which contains what to expect, what to do) set, it is the best place to do the checking as all latest data must pass through here, if any prediction doesn't meet reality[22], it can jump start a recovery routine by looking at the given decision tree[23].

If it is expected[24], select that path directly.

Else it would be an exceptional condition (totally unexpected) and the system must response to it immediately by halting all other parallel preempt calculation thread[25] then start recovery process.

Recovery process is fetch the original intent (original expectation, the goal), go to the broken branch (the branch where error occurs), asses current situation by GCA, then reformulate a new tree with the ultimate original goal as the end of the result. Then all parallel operations can go on.

This is similar to a shock, an unexpected situation suddenly bounce up and forces him to react abruptly to it, an emergency as the outcome will be unknown, terribly interrupted.
If no operation can be done to complete the original intent, select new attention point will be run again to select a new attention, it will never stop, if tree run out (PaRc success for the given branch), it will also be run as well. It is the last resort of all operation.

Contagious Memory Generation

Contagious/Consecutive memory is a series of experience chained together into one entity.
After each tree completes (PaRc pass for the decided route), create a contagious memory by linking together all the executed branches' exp data.

Rationale

Experience are ought to be joined together in order to form larger chain of action.
Select new attention point only select 1 exp at any time, and that exp can be of arbitrary length but it must be as 1 entity (exp that embeds multiple depth of exp are still considered as 1 entity).
And this is the only place that can generate sensible lengthened exp as they had been proved functional by passing PaRc.

---

21 Decided route is a continuous selected branch from the decision tree. One of the route from the tree.
22 The latest imported GCA is the reality, it contains the latest facts, the current real condition as they are all real data.
23 The selected path is the best path, and there are many alternate path that may or may not contains the situation that the system experience now.
24 Decision tree contains this clause, but is not the selected path as it is not the best.
25 As the current branch is invalidated, the further preempt branches based on this will not be accurate anymore.

How it work together

When no attention point is selected during startup, main computing system and PaRc will just be idle. ICL generate patterns from raw data input then introduce them into WM via GCA and local WM latest GCA import operation import them into local memory.

Select new attention point then runs to select new attention point for the whole system to execute. It only select one at a time to avoid confusion. Multiple selection process can be ran at the same time but only the best got to execute and be forwarded to main computing system to generate solution. Generate solution to recreate this condition as specified inside the experience[26].

Main computing system will generate solution tree accordingly with some dynamic constrain like the current globalDist, available solutions' statistic and time constrain.
It will select the best path from the tree using core decision logic, then forward that tree back into WM for execution.

WM receive that tree, disassemble it into individual exp segments marked with when to execute during tree generation and schedule them into execution loop.

When time reached, the scheduled prediction for that moment will be checked against reality by checking whether the data specified in it actually exists in the next GCA frame (scheduled to meet). At the same time any scheduled processes will also be executed by sending off signals to the relevant sensors.

If prediction meets reality, do nothing and continue.
Else raise error and prompt main computing system to give another pathway for us to continue, if he fails, start from scratch again by disposing all current tree and start over by selecting another attention point. Else continue on the path until all schedules are met.

If all goes well (PaRc passed with error but recovered or without), start contagious memory generation to group the executed route together to serve as episodic memory.
Then start from scratch again by selecting another new attention point.

This cycle will continue on forever, and it will create validated memory which can be used to predict future with variable chances of success depending on how many time it had been validated.

Episodic memory which will be the foundation of abstract thinking as it can group multitude of experience together and treat them as a single entity, thus making him able to think at higher level.

Diversified experience as he can tackle any field depending on situation and his will.

Critical thinking ability as the tree allows branch switching by design in case of error, enabling him to think critically and borrow experience from any field to help as long as they are relevant.
Most likely they will be as all experiences are built bottom up and have solid foundations.

26 A form of imitation, imitate and re-validate what had happened to improve confidence. Most important part of learning as it creates reliable component in decision making when generating decision tree, making it more likely to output what was expected, enhancing predictions.

General Processing Layer (Main computing system)

Overview

Main computing system are series of operations that cover decision tree generation, which includes experience fetching, validating, selection, rearrangement according to time, composite experience evaluation to seek of incontinence before adding it as solution.

Tasks

This list is unordered.
-Calculate and set Global Distribution value.
-Generate decision tree (solution tree) based on previous experiences for the selected attention by working memory.
-If PaRC fails, switch to the next best solution from the solution tree initiated by working memory via message or flag, if no further solution available, generate a new tree with the same goal.
-Post selected route (decision branches from the tree, aka predictions) to WM.

ICL and GCA tasks can also be grouped under this category.
So there is only 2 categories of task, one for WM and one for here.

Rationale

These operations can be treated as subset of WM's algorithms.
They are inherently related, in fact they are subsequent operations of WM.
Separated them from WM as these are the gray area functions.

All the function in WM are clearly related to it and stated clearly in academic papers[27].
Functions here are instead, hybrid of multiple unrelated operation from distinct brain region or are simply not well understood, or unproven.
It can only be vaguely categorized, thus placed them all here.

They are all indeed main functionality of brain concluded as people actually exhibits them.

These are prototyping glue created to complete the whole system flow to attain intelligence.

27 Serious research had already been done into this topic. The definitions and functions are well defined, simply search the term and it will yield surprising amount of result.

Global Distribution Update

For every operation that consist of raw data, patterns or exp generation, every single one of them will contain a polyVal describing the average inclination of their residing data.

Types

For raw data, it would be the mean of the data[28].

For patterns, it would be the mean of the extracted pattern, calculable as they are subset of original raw data[29].

For exp, it would be the average of all components' polyVal.
Example if it contains raw data, then use that raw data' polyVal as its polyVal;
If contains other exp, use that exp' polyVal as polyVal;
If contains multiple elements, sum their polyVals up and use their average as polyVal.

Timing

Update the globalDist value once for every new GCA frame arrival.
Calculate the polyVal of each GCA frame and add them to a data queue.
Update the globalDist by averaging all the data in the queue.

The queue can be of arbitrary length. Let *N* be the length of the queue.
The *N* should be a value that is simply appropriate, it has no absolute value.

For example given length of 50 GCA frames per update where its frame is 10ms apart (500ms).
If globalDist is high, shorten the length (nervous).
If it is low, lengthen it (numb).
Else if it is in between, do nothing and maintain the original threshold.

If not enough space, remove the head of the queue and allow it to enter.
Else just add it in without removing anything.
The size of the queue should never exceed *N*, but below is acceptable.
'It' here refers to the newly arrived GCA frame.

These slight modification are done to follow the dynamic nature of the protocol itself[30].

This is to ensure fairness, else the reaction may be too sharp and volatile without the data queue to smoothen the flow[31].

---

28 Mean of raw data for images means averaging all pixel values, for audio sum all points and average them, for analog sensors just the input itself. Then convert all these data into polyVal format.

29 Mean calculation method is same as raw data.

30 Nothing in the world is permanent, thus allowing it to modify its own bound may produce delicate result previously unaware of.

31 GCA although in theory should be consistent in what data they receive at any particular time, error are deemed to occur due to delays, missing or simply disconnection, leaving some part of the vital information loss, thus causing the globalDist to tremble up and down as the delayed data suddenly pop in. With data queue based averaging, this gives allowance to these problems, making it more stable and tolerant to technical related unintentional surges.

Decision tree

Structure

Separated into 2 logical node, namely main and secondary convergence.
Convergence is a concept similar to GCA, where it hold edges to other vertexes, but for here, those edges purposes can either be requirement or result or process, depending on ordering and their internal representation (be of type experience requirement, result, process or ICL patterns?).

Convergence main contains a definite, one only requirement, and have outgoing edges to multiple solutions generated based on experiences encapsulated inside another convergence vertex, namely secondary convergence.
For convergence main requirement and result are the same.
The thing given to him is what he has to do (requirement), and it is also the expected that he will do it (result). What he do is the expected result of what he has to do (requirement).

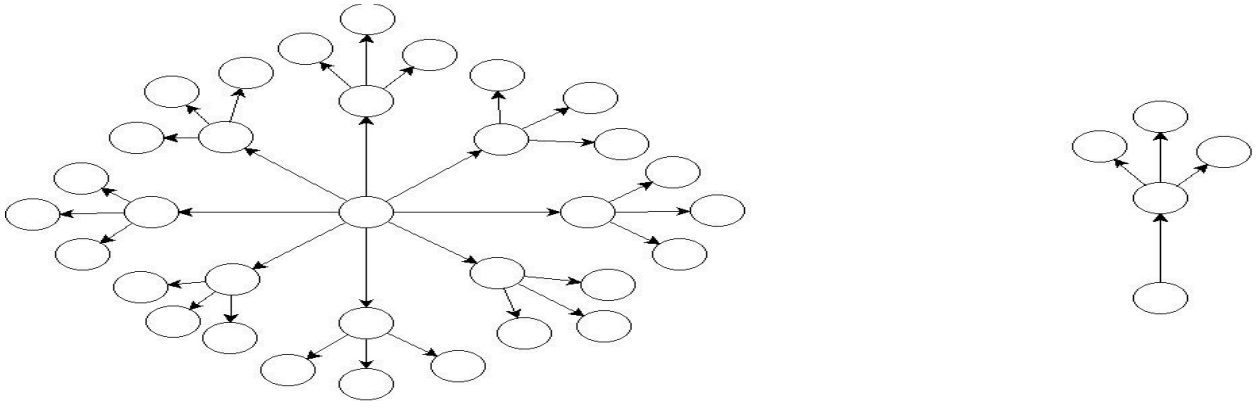
Secondary convergence contains only a single solution, where that solution is of type built upon on experience component, raw data or ICL patterns.

A solution contains requirement, result and process (what to do).
Requirement is what condition is required to meet before the result can be met by executing, if any expected processes (physical operation) and check availability of requirements.

Requirement itself is given by its parent convergence main during the creation of this secondary convergence.

Naming rationale for the term convergence

Convergence means join, group. And here it group solutions, a place where solutions converges and together they looks like a tree of converged nodes with a same goal to resolve the grand requirement (selected attention point) recursively.



Left hand side is a solution tree with 8 branches of solutions, alongside with 3 sub solution (requirements) branches. It has to be fulfilled from child to parent. Center is the selected attention.

Right hand side is one of the selected branch from the main tree.

Generation

For each requirement of secondary convergence, if the requirement is already met, then it will do nothing as it had already been done.

Else it will attempt to generate further branches by creating a new child convergence main (notice the recursion) for each of its unmet requirements. Then the child main convergence will create more solution for its assigned requirement based on experience, then more secondary convergences for each of its solution, and so on, creating the structure of a recursive solution tree.

The generation will end if all the requirements are met.
By definition the criteria for any requirement to be met are:
-It currently exist in the WM STM space, meaning it is fresh and most probably is still valid[32].
-Scheduled to be valid by decision tree logic (each requirement he made branches downward are actually an action to schedule what to do next, and as long as he follow this itinerary, all the condition should be met according to his expectation, thus can be treated as 'will be met' and get over with that.
The most inner depth of the tree are guaranteed to be raw data as it is the most fundamental element and must be met using the first criteria.

32 WM Short term memory (STM) space is the data queue which contains all the imported GCA data. By freshness it means if it has just happened moment ago, it does happened, and is probably still valid to assume that it can be used without error. It is just assumption made by the system, if it has dramatically changed within such short time frame, it will be detected as the further operation will fail as PaRc cannot pass due to wrong state. Then it will start all over again, but this time aware of that value had changed and will by then updated with latest value as well.

Branch selection and selective generation

During generation of the tree, core decision logic applies, it will fetch a series of related experience whom all of their results are the requirement of ours (completing them will result as our requirement being fulfilled).

From that list, filter and rank them according to the decision logic.
Then the best path will be preferred and encouraged to further its development, and the other solutions, will remain in the tree as backup.

He can optionally develop those branches as well in parallel, then if those branches in the end turns out to be better than the original best, they can overtake the original best.

This is possible as the decision logic is short sighted, it reviews the solution and it best suit current condition, but the actual underlying recursive solutions are not yet determined, so it may turns out to be good at the beginning, get worse and worse down the road as it was never foreseen.

To overcome this other solution can be optionally developed, but the key thing is, when the main best has finished it calculation, it will be executed without waiting for those parallel optional threads.
If they turns out to be good, they will request for a take down, then swap the attention to it.
#Currently not implemented.

Previous error will also be integrated into the solution tree so in case things goes wrong, as he already had experience on it, he can calculate them in advance, thus speed up the response time.

Thus due to the generation model, the default best branch would have already present themselves as the first solution for every branch depth.

Then it will be forwarded back to WM to execute, and if anythings goes wrong, WM will seek for premade handler (which is just other solutions down the same branch where PaRc errors occurred) and switch to it if available, else he will just call to generate a new tree.

Flow

Once received new attention point from WM, start to generate a solution tree.
When the generation is done, send the tree alongside with the selected route back to WM.
If generation failed, acknowledge WM to select a new attention point and start over again.

After returning the tree to WM, the main computing system can either be idle or generating parallel tree in attempt to seek for better solutions.

WM can raise error at any time (PaRc fail) and return us with the exact failed node and request for a branch switch to the next best solution.
If no more branches is available, generate a new tree starting from there and return that tree.
If failed to generate, acknowledge WM and wait for new attention point to arrive.

GlobalDist update are done in parallel without direct intervention with other threads except reading data input sent by them which piles up in the data queue.
The operation doesn't have to reach out to each thread to ask for data as it is expensive to tell how many threads are there and how to get to them.

Experience model

Experience design constrain

-It must be able to represent any possible combination of human cognitive process and at the same time being able to extract meaningful data from it from multiple angle.
-Its lookup speed must be equivalent or faster than human mind.
-Must be dynamically scalable.

The best storage for mind is graph.
As mind itself is also frequently being visualized as a network of neurons.
Graph can represent network, identities and links extremely well.
Traversal time is also much faster than conventional table based solution.

Any graphDB can get the job done.

The organization of data

The basic unit of graph are vertex and edge.

Constrains are:
-Exp must be able to encapsulate another exp to form a singular unit in order to be more abstract.
-Exp must contain 3 part, requirement, result and prediction.
-Exp itself can be a container for any other raw data and exp.

Experience itself is a self contained system, like a cell, it has all the information required to make something work for a very specific domain.
It contains what has to be done (requirement), what to expect (prediction) and previously executed operation (result).

Requirement

Requirement is what the system need to prepare and execute in order to meet its expected result[33].
It may contain raw data, what to be outputted to the external system and other embedded exp where the system will have to traverse downward to aggregate what the full requirements are and fulfill them accordingly.

Result

Result is the actual output after all the requirements are met.
Only set after requirement is completed.
Record the event that actually happened.

33 Expected result here means prediction. During the generation of this exp, the actual result are still unknown, thus only predictions can exist, prediction is about the same with the result, difference is that result is only set after completion of the exp, and it stores real and actual result, whereas prediction can contain flaws.

Prediction

Prediction is at that particular moment, the expected result, generated based on even previous experiences, it still remains after the whole exp has been consolidated to serve as a window to look into the past on how he made that decision for self audition purposes.

Rationale

Requirement, result and prediction are the 3 basic construct that form an exp.

For every operation in this world, there will always be a beginning, a process to go through, and an end point. Nothing is eternal once began.

For these processes, there will be certain set of requirements to be done, and only if he follow those rule, then the result can be predicted.

Exp is here to capture those rules in a crude way so that it can help in the future operation planning by making exp that is robust and accurate.

With confidence in the expectation gained from repeated exp, the operations whom system proposed will have higher chance to succeed and less surprises overtime.

Exp are accumulated from scratch and gain more and more insights from the outside world.
It is a painfully slow process, but once learned, it is permanent.

Any exp can cross relate to any other exp given that there is demand[34] for it.
Thus exp is the container and building block of decision.

Requirement and result are interoperable. It is the view point that determines who they are.
A result here is the requirement for its abstract parent as those results has to be met in order for the parent to start working (hierarchy tree).

Thus the requirement here can be another's result and vice versa.

Exp vertexes are physical entity in memory.
Can be copied and moved just like normal data, nothing magical.

34 Any other operation, as long as they are related and make sense (decision tree generation doesn't fail due to this).

Core decision logic

Core considerations

-Time ran.
-Relevance against current situation by globalDist.
-Tendencies.

Inclination Tendency

In priority order, same number means same thing, by different name:
1-Shortest route seeking tendency (if time allowed, do thing slower, else always optimize)
1-Novelty seeking tendency (anything that is completely/partially new, gets higher priority)
1-Recursive breaking tendency (if certain link repeated too many time (>=2), incline toward links with less count)

2-Central tendency (inclination toward the middle value, to stabilize the system. mean value[35] changes according to influences, quantity is not known, but bigger the influence the faster it fluctuate over it to adapt. past over 1/4 then it will begin this test)

3-Familiarity fetching tendency (if everything is going perfectly fine, it is fine to repeat certain link as it is more stable)
3-Inclination toward recreating the scenario (to validate data accuracy, mostly for generated data, recreate to put into real world test to confirm links' generation correctness)

If under same depth, the one with lower run count will be prioritized.
If under same run count, the one with lower depth will be prioritized.
This is to make sure foundation is solid before moving to higher complexity.

Used to arrange solution tree during its generation and selection phrase.
Also for all operation where it mentions “use core decision logic” at their definition.

All these inclinations are the general pattern of all decision making process, extracted and refined to ensure it will eventually lead to intelligence given enough time.

35 Mean value is globalDist, a type of polyVal.

Decision Flow

Decision logic is universal, can be used under any occasion to make reasonable judgment.
Decision is used only during path selection.

Used extensively throughout the system, the decision logic are just sets of constrains that need to be followed across all operation that wanted to make decision.
Each operation implement their own version of decision logic as the input are different thus need to be tailored independently.

For each pathway, it is required to have a polyVal indicating its mean (inclination).

Compute difference of globalDist[36] against each path's polyVal, the less difference the better.
It signifies that it is more relevant to the current trend globally.

Depending on current globalDist, current inclination[37] will change accordingly.

If globalDist is low (<25):
Select path with higher time ran (more confidence) thus taking less risk.
Select operations that are closer the middle (50) in attempt to guide the global trend back into normal range.
Route selection will be passive and doesn't care much about speed (time required for operation to complete).

If globalDist is high (>75):
Similar as globalDist is low EXCEPT
Route selection will be more aggressive and tends to rush thing (select the operation whom time required are least aggressively).

If globalDist is normal (25 ~ 75):
Select unique path to try out new things.
Select operation whom polyVal are in between its range and alternate it[38].
Route selection is neither aggressive nor passive, it just goes with the mood.
Doesn't care much about time as long as it doesn't seems to be overly high or low.

For each of those inclination defined above, they are calculated independent of each other.
Each path will be ranked for each of those inclination.
Thus it will output several array of data where each of them are ordered best at front.
Then calculate individual path's score by adding up their individual score (position, index in the array) together and rearrange them.

The final output array will be the best solution in front and least ideal solution at the end.

36 GlobalDist is available at any time on demand, and will only be the latest (never return old value).
37 Inclination are
38 Select higher than nominal, then lower than nominal, then keep repeat if order to create a wave like selection scheme to reduce the risk of over reacting which will lead to crossing the boundary of high or low. Alternating is the best way to maintain the globalDist close to nominal value (50 globalDist).

General Flow and Final Expectations

To guarantee learning potential, the system must make diligent choice between making novel actions or follow the usual route.
ISRA is like all form of life, follows the law of the Nature.

Curiosity

Curiosity is the key to attain intelligence.
It must have to motivation to reach out to things, and at the same time learn from it.
Learns from generic patterns.

Learning has 2 goals:
-Deplete unknown source to make everything predictable.
-Ability to recreate it whenever needed in order to chain up larger, more abstract operations or concept.

This leads us to:

Imitation

Imitation is the direct outcome driven by curiosity and the only way to learn.
ISRA, unlike people who comes with predefined limbic brain (instinct), he must build his own.
By imitating any patterns he came across, he will gain experiences on generic pattern and consolidate them as building blocks.

At first the experience is going to be really short and precise.
For example a few milliseconds of any audio or visual.

Here is where GCA comes in, it groups chronologically what happens in order, our unit of building blocks start from a single GCA, then keeps on expanding to more and more GCA units.

WM will select from the GCA, a small amount of data that he thinks is important and attempt to recreate it. If successful then it means it can be manipulated with at least some form of precision in the future.

Note that experience is just a subset of GCA, a smaller, confined and better controlled environment.
Slowly add more element to the exp to gain more confidence if they pass.

As more and more mini GCA elements being successfully recreated, it is possible now to expand exp to larger extends by covering more fields of GCAs and perhaps across multiple of them in time order.
As for the beginning it is not advisable to jump and joint multiple unconnected time series together until all the foundations are solid.

The exp base keeps growing and build up a foundation based on reality.
Soon the exp will be complex enough and the system can precisely predict what will happen next.
He will begin to formulate solutions to much distant future and make use of already proven exp as building blocks, attempting to joint multiple exp happened in distant into a single solution for his next selected operations.

From Imitation to Self Awareness

The select new attention point logic has a few interesting constrains:
Select the solution with greatest complexity if all other selectable attentions of lower complexity have already ran plenty of time, making them non unique, less attractive to feed curiosity.
Else resort to the one with lower complexity whom time ran is lower than the one with greatest complexity. This will ensure the foundation is rigid before moving onto more abstract concept.

Similar to what people will react to when being thrown into the wilderness alone the first time, whom all skill one learned from the modern civilization were rendered useless and one will have to resort back to one's primal instinct, as it is robust and works on most occasions.
Thus this 'resort back to foundation' instinct is given to ISRA for free.

Using this logic, ISRA can slowly begin to 'make his own decision' as soon as he ran out of things to imitate, as all of them had already been done before and all the condition is predictable.
This gives him ultimate freedom to decide what to do next on his own as he can now decide what will happen in the future with its actions based on exp with great precision.

Dawn of abstract thinking

With countless trail and error, he will eventually able to make 1 solution by imitating how people make theirs, the process will be identical, and this will be a stepping stone to larger expedition as now he can use that generation pattern to do abstract thinking.
Soon more and more solution will make sense to ISRA as he keeps on trying to cover more GCAs with high abstract content (those things which doesn't make sense until one understand it) and succeed, a model of thinking will be available to ISRA.

Personality traits

His personality will be dictated completely by the environment he lives in.
Deploy multiple instance at different location and non of them will be identical in return.
With abstract thinking set in place, he will eventually start to think about himself as after all these time, what he had done is all for himself (learning).

Self Awareness

It is a requirement for him to know about himself in order to proceed any further.
There will be times people are not around, and he do the things on his own.
He will learn from himself, what he had done, what he is doing and the most important of all, why.
Why is a general concept representable in the exp model, just as all form of exp, it is just a 'to and from' situation. A transformer that transform requirements to another requirements based on exp, that is the hidden 'why' operator in the exp operation model.

With all the time he spent alone, he will combine all the traits that he had learned and following the decision tree fetch ranking, he will incline to one of the trait he think is the best for each operation.
With more and more operations, he will settle down to the best trait he think it is, then it becomes his personal unique traits.

Deployment

At this moment, it is already possible to start servicing routine.
His mind should had matured to a level where solving cognitive problem will not pose any issue.
Common sense should already be abundant.

Can now spawn independent workers to work on real world tasks, it can leverage the global common sense library to work on any cognitive tasks that he had learned before.

Those instances may be generalized, and whether to return experiences he gained during these tasks are user dependent, they may or may not want their details to be public.

The main learning agent, however, can choose to either merge those experience learned from task, or let third party manage them and ignore it as a whole.

The learning agent can be separated completely from the user agent by composing a static library of generalized common sense library, then each of those user agent can built upon it to speed up their individual performance and customize themselves accordingly to demand.

Glossary

PolyVal

PolyVal means poly value (multiple meaning value), a 0~100 double floating point that can be represent any data type and is used system wide to represent a state.

Used to synchronize or share state among every other distinct or non distinctive operation or state.

For polyVal, 50 means nominal, the balanced point, 0 means extreme negative and 100 extreme positive.

All sensor data are required to have a transition mapping function to convert back and forth between polyVal and actual sensor data representation.

Global Distribution

A polyVal to indicate current system's general status.

This is a way to synchronize operation system wide.
All decisions and solution generating phrases uses this value to filter their current scope.

Its calculation is based on the whole system's current inclination.
If you are walking down the street and suddenly a blackout took place, you will transits from 45~75 normal operating globalDist range into <45 range, which trigger sudden shock and you body will react accordingly to protect you from any harm.

This is due to normal visual value mean (averaging pixel density) will be normally within 45~75 percentage range, but a sudden blackout it causes a sudden drop of the mean to <45 range (darkness in RGB is usually 0,0,0, thus its average mean will be low too), causing the system to alter its current globalDist accordingly.

Calculation of globalDist is simple:
Sum of all operation average mean at dynamic length of time based on configuration.
It can be 1 second of average, or 100 milliseconds of average, average means average of all operation output's polyVal.
This will ensure the globalDist to properly reflect the current ongoing trend of the whole system.

Real world example

People when they are feeling mad, their globalDist raises in order to indicate anger.
Other system in the body listen to this sync stimuli, saw that it has risen, and react accordingly.
Thus your blood pressure rises and mind tensed up.
It can only happen if this globally shared stimuli (globalDist) exist.

Global Changes Association (GCA)

All vertexes including raw data, identified pattern and all experience vertexes are required to enroll into this GCA operation.

Purposes

To bind all currently active mind elements into a single big compilation of accessible memory, so other operations can visit it and get to know other currently running operation in order to share resources and make new connections.

It also serve as a snapshot of the mind, this ease up the decision making work as it groups data into chronological order. And GCA itself can be treated as a timer of the mind, which compiles all running data into a single place and make it accessible system wide.

Only after this operation updates from unrelated sources can see each other and potentially makes new functional connection, a fundamental communication hub for all mental operations.
This make updates visible to the next operation.

For every GCA vertex instance, it will contain a time stamp to enable him to determine what operation came first and ease audit purposes as it is possible to visualize how he felt at particular moment and why by just following the GCA instances.