Fault/intrusion tolerance through Metamorphic Diversity

Felicita Di Giandomenico

ISTI-CNR, Pisa, Italy

89th IFIP WG10.4 Meeting

5-7 May 2026, Kaunas

Goal & Idea

- Evolution of redundancy-based FT with parallel executions:
  - Use of **Replicas**
  - Use of **Versions** developed through *Design Diversity*
  - ➢NEW: Use of **Versions** developed through *Metamorphic Diversity*

- **Goal**: increase the degree of **diversity** among the programs to enhance the *effectiveness* of a redundant fault/intrusion tolerance scheme
- **Idea**: to use **Metamorphic Relations** to generate either **different inputs** for the same functional specification, or different **functional specifications.** The emphasis is on **attacks.**
- **Proposal:** definition of two logical fault/intrusion tolerant architectures based on Metamorphic Diversity:
  - **NMDP**: N-Metamorphic Data Programming
  - **NMFP**: N-Metamorphic Function Programming

Methamorphic Data Diversity Relation

Metamorphic Relation (MR):

- for a given function ***g,*** an MR is expressed as a relation among a set of two or more inputs (***i, i_1, . . . ,i_n ∈ I***) and their corresponding outputs (***o_0=g(i), o_1=g(i_1), . . . ,o_n=g(i_n) ∈O***)
- Given *i* and *MR*, a set of **follow-up inputs** is derived from *i* according to *MR*

Metamorphic Data Diversity Relation (MDDR):

- ***MDDR*** = $\{R_m\}$, m=1,...,r of relations*:*
- $R_m(i, i_1(i), \ldots, i_{n-1}(i), g(i), g(i_1(i)), \ldots, g(i_{n-1}(i)) \subset I^n X O^n$, *where* $n \geq 2$, $r \geq 2$, and the elements $i_j(i)$, j=1,...,n−1, are the *follow-up* inputs derived from i according to R.
- ***Example****:* given the *function* $g(i) = sin(i)$ and a *source input* (i), an ***MDDR*** can be composed of the following metamorphic relations:
- R_1 (e.g., **sin(i) = sin(i_1)**) is defined on the inputs i and i_1, as well as the outputs $o_0 = sin(i)$ and $o_1 = sin(i_1)$, where ($i_1 = \pi - i$) is a **follow-up** input derived from i according to R_1.
- R_2 (e.g.,**− sin(i) = sin(i_2)**) is defined on the inputs i and i_2, as well as the outputs $o_0 = sin(i)$ and $o_2 = sin(i_2)$, where ($i_2 = \pi + i$) is a **follow-up** input derived from i according to R_2.

Methamorphic Function Diversity Relation

Metamorphic Function Diversity Relation (MFDR):

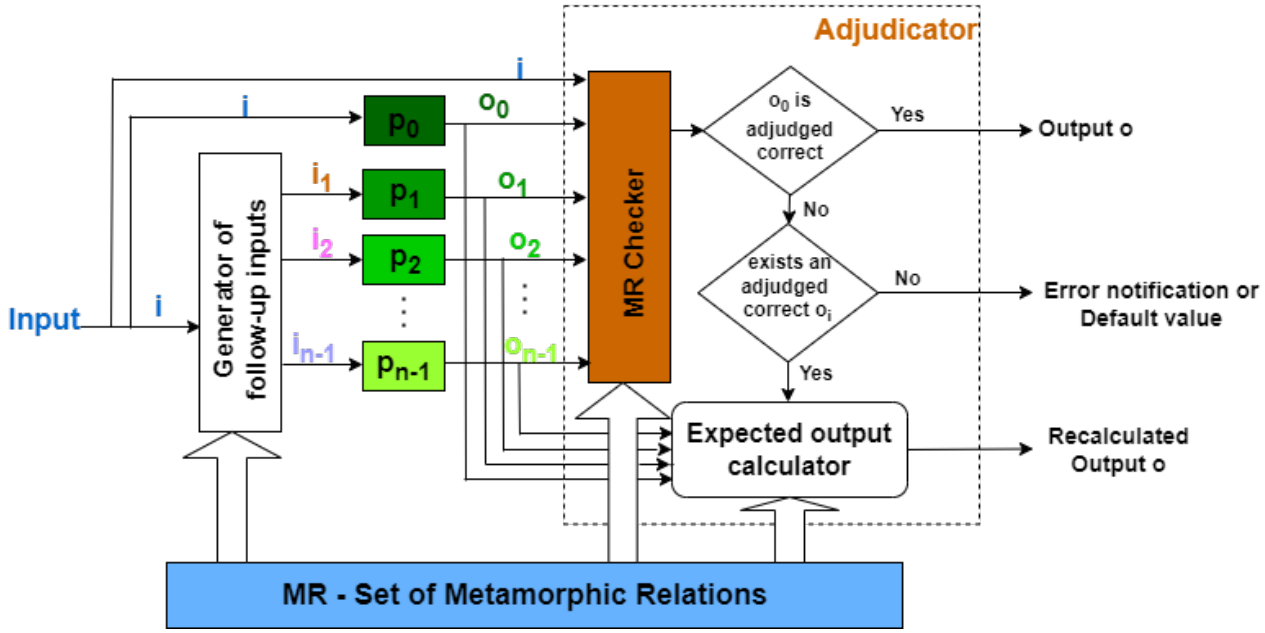
MFDR = $\{R_m\}$, $m=1,\ldots,r$ of relations:

$R_m(i, g_0(i), \ldots, g_{n-1}(i)) \subset IXOXO_1X\ldots XO_{n-1}$,

where $n \geq 2$, $r \geq 2$, and the elements $\{g_j(i)\}$, with $g_j: I \rightarrow Oj$, *are the* ***follow-up functions*** derived from $g_0(i)$ according to R.

- ***Example****:* given an existing source function (g_0 = sin), and an input (i), an MFDR can be defined using the following ***metamorphic relations****:*
- *R1* (e.g., $\sin(i)^2 + \cos(i)^2 = 1$) *is defined on sin and cos, as well as the outputs* o_0 *and* o_1 *produced by executing sin(*i*) and cos(*i*), where* g_1 *= cos is a* ***follow-up*** *function derived from* g_0 *= sin according to R1.*
- *R2 (e.g., sin(*i*) =* $\pm \tan(i)/\sqrt{1+\tan(i)^2}$*) Is defined on sin and tan, as well as the outputs* o_0 *and* o_2 *produced by executing sin(*i*) and tan(*i*), where* g_2 *= tan is a* ***follow-up*** *function derived from* g_0 *= sin according to R2.*

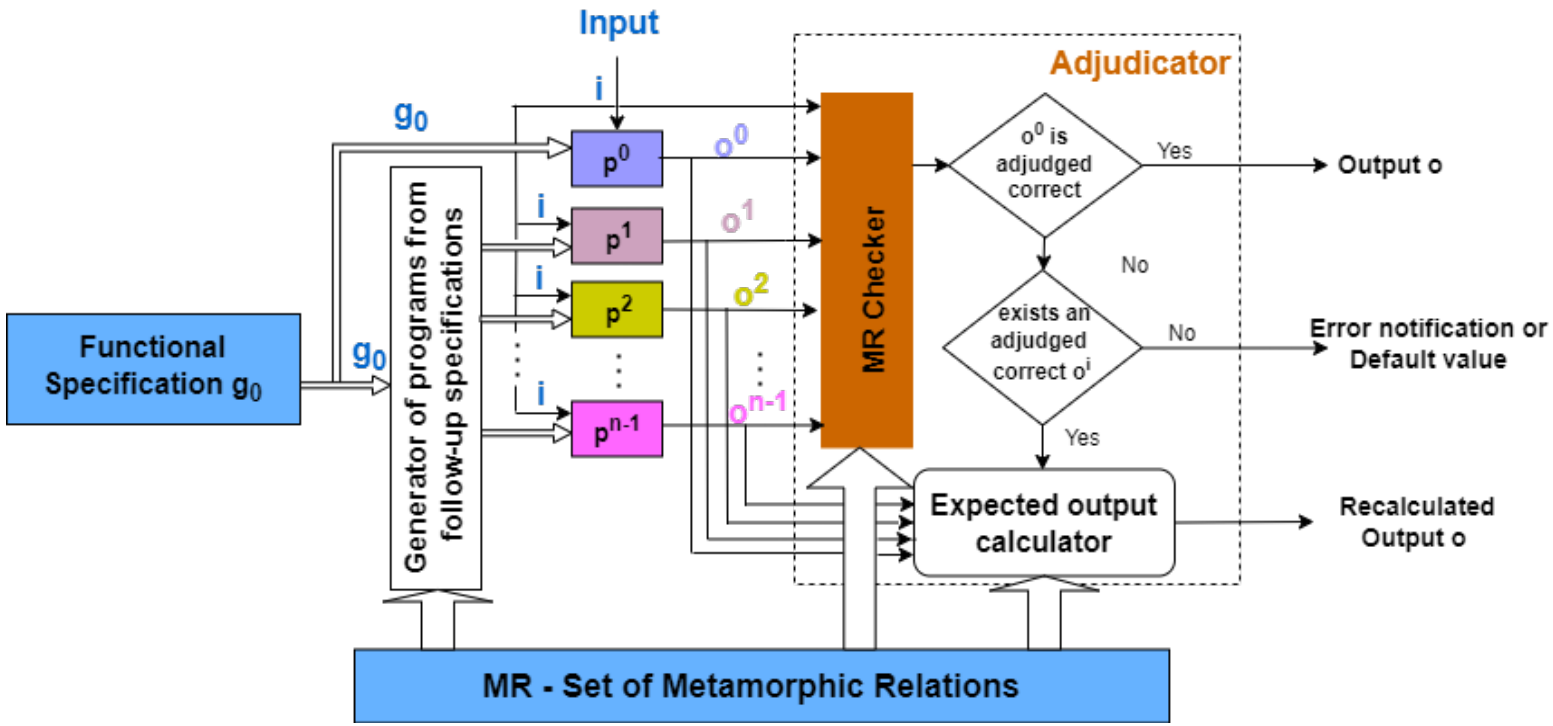
NMDP Logical architecture



$i_1, i_2, \ldots, i_{n-1}$ are different inputs derived from i according to *MR*
$p_1, p_2, \ldots, p_{n-1}$ are different versions of the program p_0

Example: given the function $g(i) = sin(i)$, the source input i , the relation *MR*: $sin(i) = sin(i_1)$, $i_1 = \pi - i$ is a follow-up input derived from i according to *MR*

NMFP Logical architecture



p^0 program implements the functional specification g_0

p^1, p^2,...., p^{n-1} are different programs implementing different functional specifications derived from *g_0* according to *MR*

Example: given the source function *g_0=sin*, the input *i*, *the relation MR:*
$sin(i)^2+cos(i)^2 = 1$, *g_1=cos* is a follow-up function derived from g_0 according to MR

Adjudication Logic

- Based on the check of couples of produced outputs, exploiting the pertinent MR

$$\mathrm{Ck} : O \times \{1, \ldots, r\} \times O \rightarrow \{0, 1\}$$

$$\mathrm{Ck}(o_i, x, o_j) = \begin{cases} 1 & \text{if } R_x(o_i, o_j) \text{ is verified} \\ 0 & \text{otherwise} \end{cases}$$

- Structured in 2 phases:
  - First, it verifies the correctness of the output o_0 (produced processing the source input i_0). If adjudged correct, it is released as final output, otherwise
  - Find an adjudged correct output o_j and from it rebuild the expected correct o_0

- To tolerate ***f*** faults/attacks, it requires a redundancy degree (***n***) and a number ***r*** of MR :

$$n = 2f + 1 \text{ and } r = \frac{n(n-1)}{2}$$

(Preliminary) Analysis of Attacker Cost

- $C_{att}\ (NVP) = c_s + (f + 1)c_i$
- $C_{att}\ (NMDP) = c_s + (f + 1)\ c_i + f\,c_r$
- $C_{att}\ (NMFP) = (2f + 1)\ c_s + (f + 1)\ c_i + f\,c_r$

where:

- c_s *is the cost to know the functional specification*
- c_i *is the cost to attack a program and manipulate its outcome successfully*
- c_r *is the cost to know a metamorphic relation between two outputs*

→ $C_{att}\ (NVP) < C_{att}\ (NMDP) < C_{att}\ (NMFP)$

Some considerations

- Is this approach worth further investigation?
- From a logical perspective, it appears promising
- **Main limitation**: the high number of MRs needed as the tolerance degree grows (with the currently developed Adjudication logic)
- A lot of work on MRs in a variety of application contexts is available from the testing community (e.g., *MT has been applied for the testing of real-world software systems such as Google and Yahoo search engines)*
    - *the expectation is that they are also useful in our context*

Full details in:

Felicita Di Giandomenico, Giulio Masetti, Francesca Lonetti, and Antonia Bertolino. *Using Metamorphic Relations in Redundancy-based Fault/Intrusion Tolerance*. ACM Transactions on Software Engineering and Methodology (October 2025). https://doi.org/10.1145/3772722