A. COMPLEJIDAD COMPUTACIONAL

A.1 Complejidad computacional

A.1.1 SumProduct

Si nos fijamos en las fórmulas de cómputo para el paso horizontal y vertical respectivamente: **Paso Horizontal:**

$$\delta q_{ml} = \underbrace{q_{ml}(0) - q_{ml}(1)}_{\text{1 suma}}$$

$$\delta r_{mn} = \underbrace{\prod_{n' \in \mathcal{N}_{m,n}} \delta q_{mn'}}_{(\bar{n}-1)\text{ x}}$$

$$\underbrace{r_{mn}(1) = (1 - \delta r_{mn})/2}_{\text{1 suma, 1 } \rightarrow} \qquad \text{y} \qquad r_{mn}(0) = (1 + \delta r_{mn})/2$$

Paso vertical:

$$\underbrace{q_{mn}(0) = \underbrace{\alpha_{mn} p_n(0)}_{\text{1 x, 1 normalización}} \underbrace{\prod_{\{m' \in \mathcal{M}_{n,m}\}} r_{m'n}(0)}_{(\bar{m}-1)\text{ x}}}_{\bar{m}\text{ x, 1 norm}} \qquad \text{y} \qquad q_{mn}(1) = \alpha_{mn} p_n(1) \prod_{\{m' \in \mathcal{M}_{n,m}\}} r_{m'n}(1)$$

$$\underbrace{q_n(0) = \underbrace{\alpha_n p_n(0)}_{\text{1 x, 1 norm}} \underbrace{\prod_{\{m' \in \mathcal{M}_n\}} r_{m'n}(0)}_{\bar{m}\text{ x}}}_{(\bar{m}+1)\text{ x, 1 norm}} \qquad \text{y} \qquad q_n(1) = \alpha_n p_n(1) \prod_{\{m' \in \mathcal{M}_n\}} r_{m'n}(1)$$

Las operaciones se resumen en la tabla A.1.

Vemos que se emplean muchos productos, que es una operación con una implementación hardware muy costosa en área.

Sumas (+)	**Productos** (x)	**Desplazamientos** (↔)	**Normalizaciones** (norm)
3	$\bar{n} + 4\bar{m}$	2	4

Tabla A.1: Complejidad Computacional del SPA

Sumas (+)	**Productos** (x)	**Desplazamientos** (↔)	**Normalizaciones** (norm)	tanh/atanh
$\bar{m} + \bar{n}$	$(\bar{n} - 1)$	$\bar{n}$	0	$\bar{n}$

Tabla A.2: Complejidad Computacional del Log-SPA

A.1.2 **Log-SumProduct (LLR-BP)**

Nodos de chequeo:

$$\eta_{m,n}^{[l]} = \underbrace{-2\tanh^{-1}}_{1 \leftarrow,\ 1 \text{ atanh}} \underbrace{\left(\prod_{j \in \mathcal{N}_{m,n}} \tanh \underbrace{\left(-\frac{\lambda_j^{[l-1]} - \eta_{m,j}^{[l-1]}}{2} \right)}_{1 +,\ 1 \leftrightarrow} \right)}_{(\bar{n}-1) \text{ tanh},\ (\bar{n}-1)+,\ (\bar{n}-1) \rightarrow,\ (n-1) \text{ x}}$$

Nodos de bit:

$$\lambda_n^{[l]} = \underbrace{L_c r_n + \sum_{m \in \mathcal{M}_n} \eta_{m,n}^{[l]}}_{1^*,\ (\bar{m}+1)+}$$

Las operaciones se resumen en la tabla A.2.

La ventaja es que al trabajar en el dominio logarítmico transformamos los productos en sumas. Sin embargo ahora tenemos que calcular funciones hiperbólicas que pueden aproximarse fácilmente con bastante exactitud.

A.1.3 **Enfoque de Gallager**

Nodos de chequeo:

$$L_{m \rightarrow n}(x_n) = \left(\prod_{n' \in \mathcal{N}_{mn}} sign(Z_{n' \rightarrow m}(x_{n'})) \right) \times \underbrace{f \left(\sum_{n' \in \mathcal{N}_{mn}} f(|Z_{n' \rightarrow m}(x_{n'})|) \right)}_{(5\bar{n}-2)\ +,\ \bar{n} \text{ exp},\ 2\bar{n} \text{ log}}$$

Sumas (+)	**Productos** (x)	**Exponenciales** (exp)	**Normalizaciones** (norm)	Logaritmos (log)
$5\bar{m}+\bar{n}-1$	1	$\bar{n}$	0	$2\bar{n}$

Tabla A.3: Complejidad Comp. Enfoque de Gallager.

Sumas (+)	**Productos** (x)	**Comparaciones**	**Correcciones LUT/Lineales**
$\bar{m}+12\bar{n}-23$	1	$3(\bar{n}-2)$	$3(\bar{n}-2)$

Tabla A.4: Complejidad Computacional del Enfoque Jacobiano.

Para el cálculo hemos tenido en cuenta la expresión de $f(x)$. Solo hemos tenido en cuenta el cálculo de la magnitud ya que el signo no requiere muchos cálculos, solo tenerlo en cuenta en las operaciones (el signo se guarda en un bit, que se irá invirtiendo o no según sea el caso).

Nodos de bit:

La expresión para los nodos de bit es la misma, 1 producto y $(\bar{m}+1)$ sumas.

A.1.4 Enfoque Jacobiano

Como ya se dijo en el apartado 2.3.3 del enfoque Jacobiano, mirando las ecuaciones 2.26 y 2.27 vemos que la función $L(U \oplus V)$ puede realizarse con 4 sumas, una comparación y 2 correcciones (mirando la LUT o con una evaluación lineal) y en cada nodo de chequeo la operación se repite $3(\bar{n}-2)$ veces. Además, se siguen empleando las mismas operaciones para los nodos de bit. En la tabla A.4 se recogen las operaciones necesarias.

A.1.5 Minsum y sus correcciones

- **Minsum**

Nodos de chequeo:

$$L_{m\to n}(x_n) = \left(\prod_{n'\in\mathcal{N}_{mn}} sign(Z_{n'\to m}(x_{n'}))\right) \times \underbrace{min_{n'\in\mathcal{N}_{mn}}|Z_{n'\to m}(x_{n'})|}_{\bar{n}-1\ +,(\bar{n}-2)\ \text{Comp}}$$

Nodos de bit:

Sumas (+)	**Productos** (x)	**Comparaciones**
$\bar{n}+\bar{m}$	1	$(\bar{n}-2)$

Tabla A.5: Complejidad Computacional del Minsum

Sumas (+)	**Productos** (x)	**Desplazamientos** ($\leftrightarrow$)	**Normalizaciones** (norm)	**Correcciones**
$\bar{m}+\bar{n}$	$(\bar{n}-1)$	$\bar{n}$	0	$\bar{n}$

Tabla A.6: Complejidad Computacional de la simplificación LUT/PW

La expresión para los nodos de bit es la misma, 1 producto y $(\bar{m}+1)$ sumas.

Las operaciones se resumen en la tabla A.5. Si usamos el minsum normalizado tendremos que realizar además una normalización en los nodos de chequeo y en el caso de la corrección por offset tendremos que realizar una comparación y una suma.

A.1.6 Aproximaciones LUT y PW

La complejidad computacional de estas dos simplificaciones será la misma que la versión ideal del algoritmo LLR pero en lugar de calcular la tanh se realizará una corrección usando una LUT o una aproximación lineal. Las operaciones se han recogido en la tabla A.6

A.1.7 Modificación del enfoque de Gallager por sustitución de la exponencial

Nodos de chequeo:

$$L_{m\to n}(x_n) = \left(\prod_{n'\in\mathcal{N}_{mn}} sign(Z_{n'\to m}(x_{n'}))\right) \times \underbrace{\Psi\left(\sum_{n'\in\mathcal{N}_{mn}} \Psi(|Z_{n'\to m}(x_{n'})|)\right)}_{(6\bar{n}-1)\ +,\ 2\bar{n}\ \log_2,\ 4\bar{n}\ \leftrightarrow}$$

Nodos de bit:

La expresión para los nodos de bit es la misma, 1 producto y $(\bar{m}+1)$ sumas.

Las operaciones se resumen en la tabla A.7.

Sumas (+)	**Productos** (x)	**Desplazamientos** ($\leftrightarrow$)	Logaritmos binarios(log_2)
$\bar{n} + \bar{m}$	1	$4\bar{n}$	$2\bar{n}$

Tabla A.7: Complejidad Computacional de la Simplificación Binaria