This paper talks about the SAGA Algorithm, its implementation, comparison with SGD, and convergence. SAGA is an optimization method and a kind of incremental gradient algorithm with a Fast linear convergence rate that supports composite objectives and non-strongly convex problems. The algorithm is defined as:

$$w^{k+1} = x^k - \gamma \left[ f'_j(\phi_j^{k+1}) - f'_j(\phi_j^k) + \frac{1}{n}\sum_{i=1}^{n} f'_i(\phi_i^k) \right],$$
$$x^{k+1} = \text{prox}_\gamma^h \left( w^{k+1} \right).$$

The paper then gives us a comparison of SAGA with other optimization techniques:

| | SAGA | SAG | SDCA | SVRG | FINITO |
|---|---|---|---|---|---|
| Strongly Convex (SC) | ✓ | ✓ | ✓ | ✓ | ✓ |
| Convex, Non-SC* | ✓ | ✓ | ✗ | ? | ? |
| Prox Reg. | ✓ | ? | ✓[6] | ✓ | ✗ |
| Non-smooth | ✗ | ✗ | ✓ | ✗ | ✗ |
| Low Storage Cost | ✗ | ✗ | ✗ | ✓ | ✗ |
| Simple(-ish) Proof | ✓ | ✗ | ✓ | ✓ | ✓ |
| Adaptive to SC | ✓ | ✓ | ✗ | ? | ? |

The need to introduce such an algorithm is to overcome the drawbacks of SGD which are that the variance of SGD's update direction can only go to zero when decreasing step sizes are used thus preventing a linear convergence rate as in GD. This can be solved by a variance reduction process instead to get a constant step size. One way to achieve this is using saga:

$$\text{(SAGA)} \qquad x^{k+1} = x^k - \gamma \left[ f'_j(x^k) - f'_j(\phi_j^k) + \frac{1}{n}\sum_{i=1}^{n} f'_i(\phi_i^k) \right],$$

We then see a comparison of SAGA vs SVRG. SAGA updates $\varphi j$ value each time j is picked while SVRG updates all $\varphi$ as a batch. Compared to SAGA, SVRG saves memory but consumes more computing power. SVRG has an additional parameter to be tuned, In NNs, however, memory is more expensive.
We are also provided with some implementation techniques that help use this more effectively:
When derivatives are sparse, you may want to update x in time so that sparse updates are done at each iteration. SAGA assumes that each $fi$ is strongly convex, but in most cases, we have only convex $fi$. We introduce the quadratic regularization term $\mu/2||x^2||$ to maintain strong convexity, and SAGA formula should be changed:

$$x^{k+1} = (1 - \gamma\mu)\, x^k - \gamma \left[ f'_j(x^k) - f'_j(\phi_j^k) + \frac{1}{n}\sum_{i} f'_i(\phi_i^k) \right]$$

As if the regularization term is separated into each $f_i$.
We then see some experimental results with the following conclusions:

1. Finito(perm) performs the best on a per epoch equivalent basis, but it can be the most expensive method per step.
2. SVRG is similarly fast on a per epoch basis, but when considering the number of gradient evaluations per epoch is double that of the other methods for this problem, it is middle of the pack.
3. SAGA can be seen to perform similarly to the non-permuted Finitocase, and to SDCA.
4. SAG is slower than the other methods with constant step size.

**References:** *Defazio et al. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. NeurIPS 2014.*