Abstract—Grammar-Guided Genetic Programming is already outperforming humans at creating efficient transmission schedulers for large heterogeneous communications networks. We have previously proposed a multi-level grammar approach which achieved significantly better results than the canonical Grammar-Guided Genetic Programming approach. Initially, a restricted ‘small’ grammar is utilised in order to discover suitable structures. A full grammar is then adopted after this initial phase. Hence, evolution can focus on maximising performance, by fine-tuning the well-structured models. In this work, we propose to use a hierarchical approach by employing multiple small grammars instead of a unique small grammar at the lower level, in conjunction with the full grammar at the upper level. To use multiple small grammars while maintaining the same computational budget, we have to use either (i) reduce the number of generations, or (ii) reduce the size of the population for the evolution with each of the small grammars. In this work, we confirm that the hierarchical grammar approach using the division of number of generations strategy achieves significantly better results than the multi-level approach, but requires defining an ideal number of small grammars to achieve the best performance. We also show that the hierarchical grammar approach using the division of population size strategy achieves significantly better results than the multi-level approach. However the division of population size strategy is less sensitive to the number of small grammars.

Index Terms—Genetic Programming, Telecommunications, Hierarchical Grammar-Guided Genetic Programming, Heterogeneous Network.

I. INTRODUCTION

The number of mobile phone users is currently estimated at 5 billion [1] and it is increasing at a fast pace. Network providers attempt to ‘loyalise’ their existing clients and sign-up additional ones through improved services and innovative technology capabilities. While attracting new clients might increase the profitability of the network providers, the additional demand for data affects negatively the overall Quality of Service (QoS) [2] that is experienced by the clients. Network operators attempt to improve the QoS of their infrastructure using efficient consolidation software (e.g., antenna duty cycle, and signal strength variation) with elaborate optimisation algorithms. Network operators also expand their physical communication infrastructure with better-performing cells [3], thus leading to an infrastructure heterogeneity.

Historically, cellular networks consist of a set of Macro Cells (MCs) to provide User Equipments (UEs) such as smart-phones with communication capability. However, the increase in number of phones rendered MCs incapable of providing an adequate quality of service. To reduce the load on MCs, Small Cells (i.e., SCs, low-powered cells) have been deployed in the field to absorb parts of the workload, thus creating a Heterogeneous Network (HetNet). SCs are often installed in areas with high network traffic (e.g., parks) to provide communication capabilities to near-by UEs. This reduces congestion on overloaded MCs. SCs have the benefit of being small and cheap devices which allows their installation in an ad-hoc fashion. However, SCs transmit at a low power, making them vulnerable to severe interference from other cells that share the same spectrum. To alleviate inter-cell interference, the 3rd Generation Partnership Project standardisation (3GPP [4]) includes a mechanism called Almost Blank Subframes (ABSs). Note that a ‘subframe’ is a one millisecond unit of time in which cells transmit data. MCs are muted during ABS sub-frames and are only active during non-ABS sub-frames. Muting MCs for a given time reduces the interference at the SCs and enables them to communicate more effectively with their attached UEs.

HetNets are complex systems and they must be continually reconfigured in real-time as conditions in the environment change. In this paper, we focus on the coupled problems of optimising the ABS patterns of MCs and scheduling communications between SCs and their attached UEs. Lynch et al. [5] were the first to put forward an autonomic algorithm, i.e., Grammar-Guided Genetic Programming (G3P) to optimise HetNets communication scheduling in a millisecond timescale. G3P evolves an expression that maps network statistics to a transmission schedule. While G3P has been shown to outperform human-designed policies, the authors also identified a remaining optimality gap that could be closed in future work. In a former work [6], we proposed using G3P with a multi-level grammar approach to gain a performance improvement. For this end, we created various grammar levels starting from a small grammar with only a restricted set of terminals (the most important ones), to a full grammar with all the available terminals. SCs are only active during non-ABS sub-frames. Muting MCs for a given time reduces the interference at the SCs and enables them to communicate more effectively with their attached UEs.

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

Authorized licensed use limited to: University College Dublin. Downloaded on December 22,2020 at 11:12:06 UTC from IEEE Xplore. Restrictions apply.
In the current work, we confirm the advantage of using a hierarchical grammar (with two levels): multiple small grammars instead of a unique one at the lower level, and one full grammar at the upper level. The idea is to (i) independently run G3P with each of the small grammars for a few generations to evolve different structurally interesting individuals, (ii) gather the best-obtained individuals from each independent run, and (iii) evolve them using the full grammar for the rest of the evolution to maximise the fitness function.

Our current work is an extension of [7] which only looked at one strategy for implementing the hierarchical grammar approach. When considering multiple small grammars at the lower level, some compromise has to ensure a fair comparison between each approach: either (i) reduce the number of generations when evolving the population of each small grammar, or (ii) reduce the size of the population and maintaining a large number of generations for each small grammar. We show in this work that both strategies significantly outperform the original multi-level approach. However, the strategy (i) requires using an adequate number of small grammars to achieve the best performance—which is not easy to define and also differs from one instance to another, whereas the strategy (ii) is less sensitive to that parameter.

Several works have previously been proposed to improve the quality of the initial population of evolutionary algorithms with greedy approaches (e.g., [8], [9]) or the quality of the final solution through local searches (e.g., [10], [11], [12]), our work is the first to investigate the use of different hierarchical grammar strategies to improve performance.

The rest of this paper is structured as follows: Section II provides a formal definition for the problem of scheduling in heterogeneous networks. Section III describes the G3P algorithm, the state-of-the-art multi-level G3P approach and our proposed hierarchical grammar approach with the two considered strategies. Section IV details our evaluation environment, whereas Section V reports and analyses the results of our experiments. Section VI concludes this work.

II. FORMAL PROBLEM DEFINITION

Let’s consider a HetNet $H$ composed of a set of MCs $\mathcal{M}$ and a set of SCs $\mathcal{S}$ with $\mathcal{M} \cup \mathcal{S} = \mathcal{C}$. Let’s also consider a set of UEs $u_i \in \mathcal{U}$ receiving a wireless signal $\sigma^f_i$ from every $c_j \in \mathcal{C}$.

A. Heterogeneous Networks

UEs often attach to the cell from which they receive the strongest wireless signal. However, given that SCs are low powered, only a few UEs attach to them on that basis.

The 3GPP standard includes a bias mechanism, i.e., Range Expansion Bias (REB) which allows increasing the number of UEs that attach to SCs. The REB enables SCs to capture UEs located in geographical positions where their signal is not the strongest. REB biases the signal $\sigma^f_i$ of $c_j \in \mathcal{C}$ to $u_i \in \mathcal{U}$ by a value $\beta_j$, with $\beta_j = 0$ for $c_j \in \mathcal{M}$ and $\beta_j \geq 0$, for $c_j \in \mathcal{M}$. Therefore, every UE $u_i \in \mathcal{U}$ attaches to a cell $c_j \in \mathcal{C}$ with:

$$c_j = \arg\max_{k=1}^{\left|\mathcal{C}\right|} (\sigma^k_i + \beta_k)$$

The geographical location in which UEs attach to a small cell $c_j$ when considering the bias $\beta_j$, but do not attach to $c_j$ without considering the bias $\beta_j$ is called the ‘Expanded Region’ $E_j$ of a SC $c_j \in \mathcal{S}$. A UE $u_i$ belongs to $E_j$ of $c_j \in \mathcal{S}$ if:

$$c_j = \arg\max_{k=1}^{\left|\mathcal{C}\right|} (\sigma^k_i + \beta_k) \quad \land \quad c_j \neq \arg\max_{k=1}^{\left|\mathcal{C}\right|} (\sigma^k_i)$$ (2)

SCs share the same wireless channel as MCs. Therefore, combining SCs and MCs maintains the network spectrum and does not necessitate either radical network upgrades or new safety regulations/permits. However, besides these advantages, sharing the same channel generates substantial cell-edge interference in the expanded regions. To mitigate this interference, the 3GPP framework divides time into frames $F$ which contain 40 sub-frames (SFs) of 1 ms duration each. Using the time domain and the ABS mechanism, network allocators can mute MCs at given SFs, and allow SCs to communicate with their attached UEs without suffering massive interference from MCs. However, while UEs at expanded regions experience a reduction in interference when muting MCs, UEs attached to MCs are not receiving any data.

Figure 1 shows a Heterogeneous Network with 1 MC, 1 SC and 21 UEs. In Subfigure 1, the SC with its low signal only attaches a few UEs, whereas most UEs attach to the MC. In Subfigure 2, we see the REB mechanism at work. The SC connects more UEs by expanding its region and offload the MC. However, at the same time, the REB introduces substantial interference in the expanded region of the SC. In Subfigure 3, the ABS mechanism is introduced and mutes the MC at the given sub-frame. Therefore, SC’s expanded region no longer experiences the interference from the MC, but the MC cannot communicate with it attached UEs.

B. Scheduling in Heterogeneous Networks

Let’s consider that a UE $u_i$ is able to download an amount of data with downlink $R_i^f$ during the SF $S_f$. This downlink rate $R_i^f$ is well-approximated by Eq., (3) using the bandwidth $B$, number $N_f$ of UEs communicating at the given SF $S_f$ and the Signal to Interference and Noise Ratio (SINR).

$$R_i^f = \frac{B}{N_f} \times \log_2 \left(1 + \text{SINR}_i^f \right)$$ (3)

MCs have a strong signal, which makes their attached UEs experience high SINR and provides them with high downlink
rates whenever the MC is not overloaded. Therefore, all UEs attached to MCs could be scheduled for transmission during all SFs at which the MCs are active, making their scheduling trivial. On the other hand, SCs are low powered devices, making UEs that are attached to them experience a relatively weak signal. Additionally, UEs attached to SCs would be subject to considerable interference by MCs during their active SFs.

The bandwidth is hard to improve as it is a scarce and expensive resource. This leaves two levers to act on (i.e., $SINR_f$ and $N_f$). We could improve $SINR_f$ of UEs attached to SCs by more often muting MCs. While this would lead to a higher $SINR_f$ for UEs attached to SCs, it also penalises the downlink rate of UEs attached to MCs (which may be more numerous). We could also attempt to reduce the number of UEs attached to SCs and communicating at the same SF. This would improve the downlink rate for the scheduled UEs. However, it would also penalise the non-scheduled UEs.

All these aspects make transmissions scheduling in HetNets a non-trivial problem. We require a system that defines both the SFs at which MCs are muted and schedules the SFs at which UEs communicate.

C. Fitness Function

HetNets operators often aim to optimise the fairness of experienced average downlink rates by all UEs [13] that is expressed in Eq., (5) as it improves low average downlink rates and does not reward high downlink rates. Fairness is the fitness function we aim to optimise and the one optimised by works on which ours is based [5], [6].

$$Fairness = \sum_{u \in U} \log (R_i)$$  \hspace{1cm} (4)

where, $R_i = \frac{1}{|F|} \sum_{S_f \in F} R^f_i$  \hspace{1cm} (5)

III. PROPOSED APPROACH

In this section, we present details of the G3P algorithm for scheduling in HetNets [5], the multi-level grammar approach to G3P [6], [14] and our proposed approach (i.e., the hierarchical grammar approach) with its two different implementation strategies (i.e., division of number of generations strategy, and division of population size strategy).

A. Grammar-Guided Genetic Programming

G3P has been used to address different problems (e.g., traffic assignment in urban areas [15], software configuration [16], and link allocation in 5G Networks [17]). The first G3P algorithm that was brought to the problem of scheduling in HetNets is by Lynch et al. [5] as implemented in the PonyGE 2 framework [18]. G3P evolves an expression according to a unique grammar $F$ in a Backus-Naur Form (BNF). The grammar $F$ includes arithmetic production rules that are common to the GP community. Additionally, it includes networking statistics as a means to incorporate domain knowledge.

\[
<expr> ::= <reg> | <reg> | <reg> | <Terminal>
\]

The majority of the production rules mentioned in the above grammar are well-known to the GP community and are easy to understand. However, $<\text{statistic}>$ contains terminals from the communication network application domain. While understanding the meaning of each terminal is not critical for the understanding of this work, we briefly describe each of them in Table I.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>downlink</td>
<td>Amount of data (bit/s) that could be transferred in a unit of time</td>
</tr>
<tr>
<td>num_variable</td>
<td>The noise is too high for a UE to communicate</td>
</tr>
<tr>
<td>num_att</td>
<td>Number of UEs that are attached to the Small Cell</td>
</tr>
<tr>
<td>airtime</td>
<td>Number of sub-frames at which a given UE is allowed to transmit</td>
</tr>
<tr>
<td>congestion</td>
<td>Number of UEs transmitting at the same sub-frame</td>
</tr>
<tr>
<td>avg_downlink_frame</td>
<td>Average channel quality of a UE over all sub-frames</td>
</tr>
<tr>
<td>max_downlink_frame</td>
<td>Maximum channel quality of a UE over all sub-frames</td>
</tr>
<tr>
<td>min_downlink_frame</td>
<td>Minimum channel quality of a UE over all sub-frames</td>
</tr>
<tr>
<td>avg_downlink_SF</td>
<td>Average channel quality of UEs attached to a SC at a given sub-frame</td>
</tr>
<tr>
<td>max_downlink_SF</td>
<td>Maximum channel quality of UEs attached to a SC at a given sub-frame</td>
</tr>
<tr>
<td>min_downlink_SF</td>
<td>Minimum channel quality of UEs attached to a SC at a given sub-frame</td>
</tr>
<tr>
<td>avg_downlink_cell</td>
<td>Average downlink per cell of average downlink per sub-frame</td>
</tr>
<tr>
<td>max_downlink_cell</td>
<td>Maximum downlink per cell at any given sub-frame</td>
</tr>
<tr>
<td>min_downlink_cell</td>
<td>Minimum downlink per cell at any given sub-frame</td>
</tr>
</tbody>
</table>

G3P uses the Algorithm 1 to map the evolved expressions and the network statistics to a transmission ‘interest’ every time a scheduling decision has to be made: whether or not to schedule a UE to communicate at a given SF. For each UE $u$ at every SF $f$, the expression is evaluated using the network statistics at that SF, and $u$ is scheduled providing there is a positive interest and a sufficiently high $SINR$.

B. Multi-Level Grammar

In addition to the full and more thorough grammar (i.e., $F$ as outlined above) defined by Lynch et al. [5], we have previously described a smaller and more restricted grammar (i.e., $S_f$) by only keeping a subset of terminals that we believe are the most critical [6].
Therefore, improving it would have a better impact on the initial population to \( G_3P \) using the following grammar (i.e., taken from \([6]\), we design by hand four other small grammars.

For \( \mu \) to the full grammar \( F \) from \([5]\), we define multiple small grammars for the lower level. Therefore, in addition to the example of the hierarchical grammar approach with two small grammars (\( S_1 \) and \( S_2 \)) and a full one (\( F \)), we adapted the \( G_3P \) algorithm to take the grammar \( S_1 \) at the start of the evolution and dynamically modify the grammar to \( F \) after a certain number of generations (in our case, after 10 generations).

All individuals obtained using the grammar \( S_1 \) are seeded as an initial population to \( G_3P \) using the following grammar (i.e., \( F \)).

While updating the grammar, we do not require any modification in the representation of the individuals as \( G_3P \) uses a tree representation of individuals and the grammar \( S_1 \) is included in the full grammar \( F \). The individuals also do not require the re-evaluation of their fitness as we use the same mapping algorithm and fitness function.

### C. Hierarchical Grammar Approach

In this work, we also design two grammar levels. However, unlike in the multi-level grammar approach, we design several small grammars for the lower level. Therefore, in addition to the full grammar \( F \) from \([5]\), we define multiple small grammars \( S_i | i \in \{1, \ldots, \mu\} \). In our case, we decided to design 5 small grammars (i.e., \( \mu = 5 \)) to showcase our proposed architecture, but more refined tuning needs to be performed in that regard in the future. While the small grammar \( S_1 \) is taken from \([6]\), we design by hand four other small grammars \( S_2, S_3, S_4 \) and \( S_5 \) in a similar way as \( S_1 \) by varying their terminals. All \( S_i | i \in \{2, \ldots, 5\} \) are a subset of \( F \) and their production rules \( <\text{const}> \) and \( <\text{statistic}> \) have between 2 and 4 terminals each:

\[
\begin{align*}
S_2 & <\text{const}> ::= 0.1 | 0.4 | 0.7 | 1.0 \\
S_3 & <\text{statistic}> ::= \text{downlink} | \max_{\text{downlink frame}} \\
S_4 & <\text{const}> ::= 0.0 | 0.2 | 0.4 | 0.6 \\
S_5 & <\text{statistic}> ::= \text{downlink} | \text{min}_{\text{downlink cell}} | \max_{\text{downlink frame}} | \text{min}_{\text{downlink cell}}
\end{align*}
\]

Note that \( <\text{statistic}> \) rules always contain the terminal ‘downlink’ as it is the most important statistic \([6]\) (we try to improve the downlink). In addition, we include one to two other relevant terminals from the set \( \{\text{min}_{\text{downlink frame}}, \max_{\text{downlink frame}}, \text{min}_{\text{downlink cell}}\} \) that have been shown to have an impact on the fitness function \([5]\). Rules \( <\text{const}> \) are designed to cover different parts of the search range (whole, centre range, higher range, and lower range).

In our work, we investigate two different strategies for implementing the hierarchical grammar approach depending on: (i) population size \( \text{PopSize} \), (ii) number of generations for the lower level if we only use one small grammar \#\( \text{LowLevelGen} \), and (iii) number of small grammars \#\( \text{SmallGrammars} \):

- Dividing number of generations: we generate a population of size \( \text{PopSize} \) for each of the small grammars and evolve each of them individually for \#\( \text{LowLevelGen} \) generations.
- Dividing population size: we generate a population of size \( \frac{\text{PopSize}}{\#\text{SmallGrammars}} \) individuals for each of the small grammars and evolve each of them individually for \#\( \text{LowLevelGen} \) generations.

1) Dividing Number of Generations Strategy: Figure 2 shows an example of the hierarchical grammar approach with two small grammars (\( S_1 \) and \( S_2 \)) and a full one (\( F \)), using the division of number of generations strategy. The computational budget for the lower level is \( x \times \text{PopSize} \), whereas the computational budget for the entire evolutionary process is \( (x+y) \times \text{PopSize} \). \( G_3P \) generates two initial populations (one using each small grammar) of size ‘\( \text{PopSize} \)’ each. \( G_3P \) independently evolves each of them for \( \frac{2}{x} \) generations with the same grammar used to generate them. Afterwards, the \( \frac{\text{PopSize}}{\#\text{SmallGrammars}} \) best individuals (in our case, \( \frac{\text{PopSize}}{2} \) individuals) from each resulting population are selected and aggregated to form the initial population with \( F \), which is then evolved using the full grammar for \( y \) generations.

2) Dividing Population Size Strategy: Figure 3 shows an example of the hierarchical grammar approach with two small grammars (\( S_1 \) and \( S_2 \)) and a full one (\( F \)), using the division of population size strategy. \( G_3P \) generates two initial populations (one using each small grammar) of size \( \frac{\text{PopSize}}{\#\text{SmallGrammars}} \) each (in our case: \( \frac{100}{2} \) individuals per initial population). \( G_3P \) independently evolves each of them for \( x \) generations with the same grammar used to generate them. Afterwards, all the \( \frac{\text{PopSize}}{\#\text{SmallGrammars}} \) individuals in the resulting population from the evolution with each small grammar are selected and aggregated to form the initial population with \( F \), which is then evolved using the full grammar for \( y \) generations.
IV. EXPERIMENTAL DESIGN

In this section, we describe the dataset, the setup and the statistical test used to assess the significance of our results.

We use in our work the same three HetNet instances as those used in the works we are comparing against, i.e., [5], [6]. All the HetNets simulate 21 MCs spread in a hexagonal pattern in a 3.61 km$^2$ area of Dublin city centre. The three scenarios, however, differ in their number of SCs. The least dense HetNet contains 21 SCs (1 SC per MC on average). The average density HetNet contains 63 SCs (3 SCs per MC on average). The densest HetNet contains 105 SCs (5 SCs per MC on average). Furthermore, 1250 UEs are considered in each of the scenarios. Each of the UEs is attached to either an MC or an SC.

We use the G3P algorithm provided by the authors [5] that is implemented using the PonyGE 2 framework [18]. We set the evolutionary parameters as shown in Table II.

We perform the non-parametric test, i.e., two-tailed Mann-Whitney U test (MWU) to check the significance of our results. MWU takes performance values (best fitness values) obtained by two algorithms from each run (in our case: 30) and returns the p-value that one algorithm achieves different results than the other. We consider tests with p-values $< 5\%$ significant.

V. EVALUATION

We would like to evaluate in this section the advantage of using a hierarchical grammar approach over both a multi-level grammar approach and the original G3P (with one full grammar). Let $S_i$ be the set of small grammars $\{S_1, S_2, ..., S_i\}$. We define the 6 configurations below:
Evolutionary parameters defined for the different G3P approaches: single grammar, multi-level grammar and hierarchical grammar.

<table>
<thead>
<tr>
<th>Initiation</th>
<th>Ramp Half-Half</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max initial tree depth</td>
<td>20</td>
</tr>
<tr>
<td>Overall max tree depth</td>
<td>20</td>
</tr>
<tr>
<td>Population size</td>
<td>100</td>
</tr>
<tr>
<td>Number of generations</td>
<td>100</td>
</tr>
<tr>
<td>Selection</td>
<td>Tournament 10% of population</td>
</tr>
<tr>
<td>Tournament size</td>
<td>Generation with elites</td>
</tr>
<tr>
<td>Replacement</td>
<td>1% of population</td>
</tr>
<tr>
<td>Elite size</td>
<td>Sub-tree (70% probability)</td>
</tr>
<tr>
<td>Crossover type</td>
<td>Sub-tree (once per individual)</td>
</tr>
<tr>
<td>Number of runs</td>
<td>30</td>
</tr>
</tbody>
</table>

- G3P with the full grammar from the start to the end of the evolution [5].
- S10F: the multi-level grammar approach [6] with G3P starting with one small grammar (i.e., S1) and introducing the full grammar at generation 10.
- S10F with i ∈ {2, 3, 4, 5}: the hierarchical grammar approach with G3P starting with i small grammars (i.e., S1,...,Si) and independently evolving a population of size p with each of the small grammars for a number of generations g, before gathering the best b individuals from each of the independent runs to create a full population that is evolved with the full grammar F for the remaining generations. The parameters p, g, and b are set to different values depending on the used strategy.

Note that depending on the strategy used with the hierarchical grammar approach, S10F with i ∈ {2, 3, 4, 5} will take different values for each of the parameters p, g, and b:

- Division of number of generations: p = \( \frac{\text{PopSize}}{i} \) and b = \( \frac{\text{PopSize}}{i} \). Division of population size: \( p = b = \frac{\text{PopSize}}{i} \) and g = 10.

In this section, we evaluate the performance of the hierarchical approach in two phases according to the considered strategies. First, we evaluate the performance using the division of number of generations strategy. Then, we evaluate the performance using the division of population size strategy.

A. Dividing Number of Generations Strategy

Figure 4 shows the evolution per generation of the best fitness on each instance, obtained by G3P when using the different grammar configurations (results are averaged over 30 runs). For hierarchical grammar configurations, the division of number of generations strategy is used.

Figure 4 shows that G3P improves the best fitness using all grammar configurations with division of number of generations strategy. Figure 4 confirms that using the multi-level grammar approach S10F outperforms the single grammar F in all instances. It also shows that the hierarchical approach S10F using the division of number of generations strategy yields a better performance over all instances (jointly with S10F on 63 and 105 SCs) than both the single and the multi-level grammar approaches. However, it also shows that other hierarchical approaches (i.e., S10F and S510F) using the division of number of generations strategy perform poorly as they are outperformed by the multi-level grammar approach in all instances and achieve worse results than the single grammar approach in most cases. This is mainly because using a large number small grammars means that G3P is only allowed a small number of generations to optimise the populations that were generated with each of these grammars (remember that the lower level has to share a computational budget of 10 generations). This is more acute in the case of S510F where each small grammar is allowed 2 generations (10 generations divided by 5 small grammars) to evolve its population.

We notice that the hierarchical grammar approach using the division of number of generations strategy could outperform the performance of a G3P algorithm and outperforms the use of a single or a multi-level grammar approach. However, the number of grammars at the lower level (i.e., number of small grammars) has to be tailored to allow G3P to evolve the population that is generated using each of these grammars. In our work, we decided to use the same number of generations allowed to the lower level as in the multi-level approach [6] (i.e., 10) to mitigate the effect of modifying this parameter and make sure that any improvement would be the result of the hierarchical grammar approach. Furthermore, we defined the 5 small grammars \( S_i \in \{1,...,5\} \) and chose to use them in a particular order (i.e., we have to select \( S_i \) to be able to select \( S_{i+1} \) for every \( i \in \{1,...,4\} \)). We anticipate that increasing the computation budget for the lower level, choosing different grammars or setting a different grammar selection order might affect the ideal number of small grammars at the lower level. In our case, we have seen that using 2 or 3 small grammars is ideal, but the most adequate number varies between instances.

Table III reports the mean and the standard deviation of the results obtained over 30 runs by G3P with the different grammar configurations using the division of number of generations strategy. It also includes the p-value between each of the
grammar configurations and either the full grammar F alone or the multi-level grammar approach S^{10F}.

Table III confirms that the hierarchical grammar approach S^{210F} using the division of number of generations strategy significantly outperforms both the single grammar F and the multi-level grammar S^{10F} approaches on all instances. It also shows that S^{310F} with the division of number of generations strategy significantly outperforms F and S^{10F} on all instances (except on 21SCs where results are not statistically significant). Furthermore, while S^{210F} with the division of number of generations strategy achieves the best overall mean results on the least dense instance (i.e., 21SCs), S^{310F} with the division of number of generations strategy achieves the best mean results on the densest instances (i.e., 63 SCs and 105SCs).

Table III also shows high standard deviations with respect to the difference in mean values. However, the standard deviation with S^{210F} is the lowest in every instance and is a sign of more stable behaviour. The standard deviation increases from S^{210F} to S^{310F} (except between S^{310F} and S^{410F} on 63SCs). Therefore, using more small grammars at the lower level either makes G3P behave more erratically (converges to different fitness values) or not fully converge in the given computational budget (requires more generations to fully converge).

B. Dividing Population Size Strategy

Figure 5 shows the evolution per generation of G3P’s best fitness on each instance, when using each grammar configuration (averaged over 30 runs). For hierarchical grammar configurations, the division of population strategy is used.

Note that while Figure 5 shows different number of generations for each of the plots, each grammar configuration is given the same computational budget. In the division of population strategy, the evolution with small grammars is performed using populations of smaller sizes, thus allowing us to run more generations with a lower computational cost. In our case, we run \(10 \times (\#SmallGrammars - 1)\) extra generations.

Figure 5 shows that G3P improves the best fitness using all grammar configurations with division of population strategy. Figure 5 also shows that running G3P with the hierarchical grammar approaches S^{210F} and S^{310F} using the population division strategy yields a better fitness than both the single and the multi-level grammar approaches over all instances. However, unlike what we noticed with the division of number of generations strategy, running G3P with the hierarchical grammar approaches S^{410F} and S^{510F} using the division of population strategy also yields better results than the single and the multi-level grammar approaches over all instances. This shows that the division of population strategy is less sensitive to the number of small grammars at the lower level.

Table IV reports mean and standard deviation of the results by G3P with the different grammar configurations using the division of population strategy (over 30 runs). It also includes the p-value between each of the grammar configurations and either the full grammar F alone or the multi-level grammar approach S^{10F}. Table IV shows that all the hierarchical grammar approaches S^{210F}, S^{310F}, S^{410F} and S^{510F} using the division of population size strategy significantly outperforms both the single grammar F and the multi-level grammar S^{10F} approaches on all instances. Table IV also shows that standard deviations with the division of population size strategy are higher than the differences in means. However, they are at the same level as those observed when using the division of number of generations.

VI. Conclusion

We have recently proposed a multi-level grammar approach to G3P which has showed to evolve highly performing transmission schedulers for Heterogeneous Networks. In this approach, G3P (i) evolves structurally interesting individuals with a small grammar at the lower level, then (ii) introduces a more thorough grammar at the upper level to investigate the full search space and evolve better performing individuals.

In this work, we investigated a hierarchical grammar approach to G3P. Instead of using a single small grammar at the lower level, G3P independently uses several small grammars. Then, G3P gathers the best individuals with each of these grammars and continues to evolve them using the full grammar. We proposed two different strategies for implementing the hierarchical grammar approach while maintaining the computational budget: dividing number of generations, and dividing the population size between small grammars.

We confirmed in this work that using the division of number of generations strategy can outperform the original G3P and the multi-level G3P approaches. However, it requires defining an ideal number of small grammars to be used at the lower level beyond which performance is significantly impacted.

On the other hand, we have shown that the hierarchical grammar approach using the division of population size strategy also outperforms the original G3P and the multi-level G3P approaches. However, the hierarchical grammar approach is less sensitive to the number of small grammars.
In the future, we aim to study the automatic grammar design and the definition of the computational budget for the lower level of the hierarchical grammar approach. Furthermore, we would like to apply our hierarchical grammar approach to different problem domains.

Acknowledgement

This research is based upon works supported by the Science Foundation Ireland under Grant No. 13/IA/1850 and Grant No. 13/RC/2094.

References


