Embedding a Long Short-Term Memory Network in a Constraint Programming Framework for Tomato Greenhouse Optimisation

van Bokkem, Dirk; van den Hemel, Max; Dumančić, Sebastijan; Yorke-Smith, Neil

DOI
10.1609/aaai.v37i13.26867

Publication date
2023

Document Version
Final published version

Published in
AAAI-23 Special Programs, IAAI-23, EAAI-23, Student Papers and Demonstrations

Citation (APA)
van Bokkem, D., van den Hemel, M., Dumančić, S., & Yorke-Smith, N. (2023). Embedding a Long Short-Term Memory Network in a Constraint Programming Framework for Tomato Greenhouse Optimisation. In B. Williams, Y. Chen, & J. Neville (Eds.), *AAAI-23 Special Programs, IAAI-23, EAAI-23, Student Papers and Demonstrations* (pp. 15731-15737). (Proceedings of the 37th AAAI Conference on Artificial Intelligence, AAAI 2023; Vol. 37). American Association for Artificial Intelligence (AAAI). https://doi.org/10.1609/aaai.v37i13.26867

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Embedding a Long Short-Term Memory Network in a Constraint Programming Framework for Tomato Greenhouse Optimisation

Dirk van Bokkem[1,2], **Max van den Hemel**[2], **Sebastijan Dumančić**[1], **Neil Yorke-Smith**[1]

[1] Delft University of Technology, Delft, The Netherlands
[2] Delphy B.V., Bleiswijk, The Netherlands
dirkvanbokkem@live.nl, s.dumancic, n.yorke-smith@tudelft.nl, m.vandenhemel@delphy.nl

Abstract

Increasing global food demand, accompanied by the limited number of expert growers, brings the need for more sustainable and efficient horticulture. The controlled environment of greenhouses enable data collection and precise control. For optimally controlling the greenhouse climate, a grower not only looks at crop production, but rather aims at maximising the profit. However this is a complex, long term optimisation task. In this paper, Constraint Programming (CP) is applied to task of optimal greenhouse economic control, by leveraging a learned greenhouse climate model through a CP embedding. In collaboration with an industrial partner, we demonstrate how to model the greenhouse climate with an LSTM model, embed this LSTM into a CP optimisation framework, and optimise the expected profit of the grower. This data-to-decision pipeline is being integrated into a decision support system for multiple greenhouses in the Netherlands.

Introduction

Global food demand is facing growing challenges, including the rapid increase of world population, climate change, and limited availability of grower expertise. Sustainable solutions in agri- and horticulture are needed to meet the future demands. Data-driven greenhouses (van Straten and van Henten 2010; Iddio et al. 2020) can play an important role in such solutions: they support higher productivity, prolonged cultivation periods, and growing crops closer to their consumption location. Moreover, data-driven cultivation, while helping to meet the increasing global food demands, can also help lower the impact of greenhouses on the climate.

While the benefits of data-driven greenhouses are obvious, they have not been readily adopted by practitioners who still largely rely on their own experience to optimise crop yield. Decision support systems (DSS) in greenhouses focus on short-term decisions that impact the greenhouse's climate. For instance, opening the windows or increasing the heating tube's temperature. Growers, however, make such decisions based on long-term economic objectives and to optimise eventual profit discounting the costs.

What is missing currently is a DSS that understands long-term economic consequences of short-term climate decisions in a greenhouse (van Straten and van Henten 2010; van Straten, Challa, and Buwalda 2000). Providing such a system is especially relevant for current times of uncertainty regarding energy and gas prices as a grower's experience alone might not suffice to make the best decisions.

This work introduces a DSS for greenhouses that optimises long-term economic profit through short-term actions impacting the climate in a greenhouse. The main challenge in developing such a system is that the effects of actions are delayed and have to be predicted in order to plan future actions. The DSS has to predict the impact of particular actions on greenhouse climate and crop growth. Greenhouse climate and crop growth models are an active area of research and existing (equation-based) models could, theoretically, be used within a DSS. However climate models are computationally complex and parameter sensitive, which prevents them from being used within a real-time decision making system. To overcome this issue, we replace mathematical climate models with those learned from data through machine learning. Crop growth models can be similarly complex, but in our study an existing simple model was used.

Integrating machine learning (ML) models into decision-making systems, also referred to as decision-focussed learning, is the main technical challenge we address. The existing work on such integration focuses on using ML to guide search in decision-making systems (Bengio, Lodi, and Prouvost 2021; Dai et al. 2017) or frames ML as a constraint satisfaction problem (Demirović et al. 2022; Hu et al. 2020; Verhaeghe et al. 2020; Goyal, Dumancic, and Blockeel 2022). Our problem is different: our decision-making system needs to use ML models to predict the impact of its decisions on the greenhouse climate and crop growth.

In summary, our contributions are as follows:

- We introduce a DSS that integrates short-term decisions and long-term economic effects;
- We describe the underlying technical framework that encodes a machine learning model, namely a Long Short-Term Memory (LSTM) neural network, into a decision-making system, based on constraint programming (CP);
- We report a case study demonstrating the benefit of the approach for greenhouse economic optimisation, and outline the lessons learnt while developing the system.

This work is being integrated into a live DSS for multiple greenhouses in the Netherlands.

Background and Related Work

Greenhouses are closed systems that provide a way for growers to cultivate crops that would normally not grow in a certain region or season (Nemali 2022). The crops are protected from extreme weather conditions and their production is optimised through various techniques, like artificial LED lighting and CO_2 injection (Morrow 2008; Rodriguez et al. 2015). Looking more in-depth at the operation of greenhouses, we arrive at energy and mass balances that model the current state of the greenhouse climate and crop and are influenced by both outside weather and actions inside the greenhouse (Rodriguez et al. 2015). Much research seeks to find these physical relations within the climate and crop (Vanthoor 2011; Stanghellini 1987; Jones et al. 1991). While these mathematical models approach reality more closely, they can be complex and often need many parameters to work, making their implementation impractical and difficult (Lopez-Cruz et al. 2013).

Machine learning models can cover the complex dynamics of the greenhouse and crop without many parameter settings or calibrations. However an adverse consequence of using machine learning models is that they work as a black box and do not provide useful knowledge on what happens in the greenhouse, which is especially troublesome in the case of extreme situations that occur outside of the domain on which such models were trained (Katzin, van Henten, and van Mourik 2022). Nevertheless, machine learning models are increasingly being used within the field of horticulture, especially in the form of time-serial deep neural networks (Ali and Hassanein 2020; Jung et al. 2020). More recent work also applies these to crop growth prediction (Lee et al. 2020; Alhnaity et al. 2020).

A particularly useful method for solving complex real-world problems, and not yet found in the literature regarding the greenhouse optimal control problem, is CP. As discussed in Wallace (1996), the most important features of CP are *declarative problem modelling*, *propagation of the effects of decisions*, and *efficient search for feasible solutions*. We deem CP useful because it provides a high-level approach of modelling our optimisation problem without manually having to implement any search heuristics. Lombardi, Milano, and Bartolini (2017) suggest the concept of combining learned relations between variables from data and embedding these relations into an optimisation model, called Empirical Model Learning (EML). In an earlier study, the same authors explore this approach by embedding a trained Neural Network (NN) into a CP algorithm, for a "temperature aware workload allocation problem", with promising results (Bartolini et al. 2011). The literature holds no embedding of LSTMs in CP, whereas the successes of both LSTMs and CP could prove to be a powerful combination.

Methodology and Implementation

Figure 1 provides a sketch of our framework. The central component of our framework is the machine-learned greenhouse model which is impacted by the outside weather O, retrieved from weather forecast, and possible actions A, such as turning on artificial lighting and opening the windows. The greenhouse model predicts the inside weather (I) which determines the crop growth (G). The profit is impacted by two things: crop growth at time t_n and the costs resulting from the selected actions. The problem to be solved is summarised as:

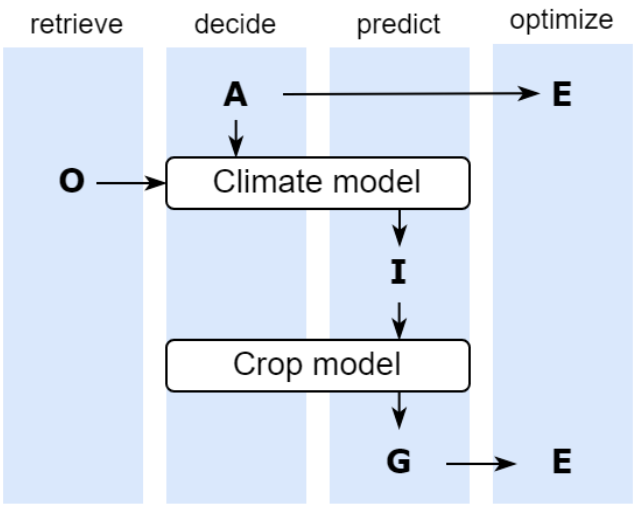

Figure 1: Overview of the CP model and its relevant parameters and decision variables. The outside weather (**O**) is retrieved from a weather forecast, the greenhouse actions (**A**) must be decided, so that the inside climate (**I**) and crop growth (**G**) can be predicted through the embedded models, finally leading to an economic result (**E**) that is used in the optimisation process for the growers

Given A set of actions A;
Time span $T = \{t_1, t_2, ..., t_n\}$;
Outside weather conditions O;
Inside climate I;
Crop growth G;
A climate model C;
A crop growth model P; and
An economic model E

Find A sequence of actions $\{a_1, a_2, ..., a_n\}$
which maximises the profit at t_n under E

In the remainder of the section, we outline our implementation. We explain how the entire framework is encoded in CP, which includes the machine learning model of the climate, a mathematical crop model, and an economic model, together with possible actions and their impact.

Greenhouse Climate Model

The greenhouse climate is modelled with an LSTM. We chose LSTM as the problem is an instance of multi-stage prediction: the climate in a greenhouse at time t depends on the outside weather, inside climate, and the actions performed at time $t-1$ (Figure 2). We consider the outside weather to be known in advance.

The main factors of the inside climate that influence the crop growth are (PAR-)light, temperature, humidity deficit, and CO_2 (Challa 1990). Different representations of humid-

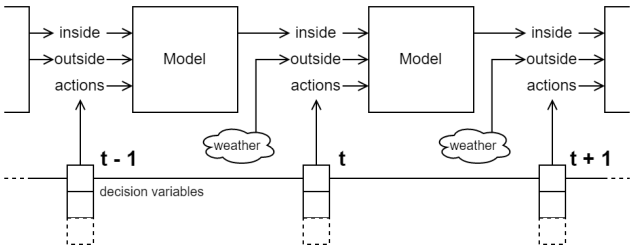

Figure 2: Multi-stage prediction in the context of greenhouse decision making. Input of each timestep is inside climate, forecasted outside weather, and decided actions of previous timestep(s). Output is inside climate for the next timestep. In this simplified example, the chosen time lag is 1 timestep. A more common use case has multiple timesteps of input.

ity exist, but we used humidity deficit because it indicates how much transpiration can occur in the crop. The important factors of the outside climate include radiation, temperature, and absolute humidity. CO_2 was omitted as feature, because the outside CO_2 level remains virtually constant. We used absolute humidity, as even at 100% relative humidity outside, moisture inside the greenhouse can be reduced by ventilation because of temperature differences (van Weel, Geelen, and Voogt 2018). Additionally wind speed plays an important role, as there is an interplay between the inside and outside climate through the windows.

Put together, the LSTM modelling in inside climate is trained on the following features: 1. *Outside weather*: radiation, temperature, absolute humidity, and wind speed. 2. *Greenhouse actions*: heating undertube, heating growtube, leeside ventilation, windside ventilation, LED lighting, SON-T lighting, and shading screen. 3. *Inside climate*: PAR-light, temperature, humidity deficit, and CO_2.

The LSTM is trained offline on the actual greenhouse data and the trained model is embedded in the overall CP framework. We implemented the LSTM in Python, using the Keras library (Chollet et al. 2015).

Crop Growth Prediction

We adopted the crop model of our industrial partner that is an adaptation of the Lintul-3 model (Shibu et al. 2010). This model calculates the fresh-weight crop growth in grams by combining the temperature, CO_2 and light components that were predicted by the LSTM climate model. While the model is relatively basic and short-term, it demonstrates the usage of a crop model in the application. More sophisticated models can be incorporated.

Embedding LSTM in CP

Achieving an efficient encoding of an LSTM into a CP model is not a straight-forward task. The main challenge lies in the conceptual difference between neural networks and constraint models. NNs are functions with well-defined inputs and outputs, which allows their efficient execution. On the other hand, constraint models are declarative: they state variables and the relationships between their values, without explicit input and output roles. The benefit of the declarative models is that variables can easily be switched from inputs to outputs without changing the model. Functions can be easily turned into relations, but not necessarily efficiently.

At its core, the LSTM cells perform various matrix operations realising the gate and state functions. We implemented the CP equivalent of these operations to represent LSTM cells, with general gate and state functions. The cell and hidden states are updated using a time lag l. At $t = 0$, these vectors c_0 and h_0, representing the initial cell and hidden states, are initialised with zeros. For each t in l, vectors c_t and h_t are computed by using the vectors of the previous timestep c_{t-1} and h_{h-1} in the before-mentioned LSTM functions. Important to note here is that in a CP approach, we cannot simply create vectors c and h and update these. We need to create vectors for each timestep, such that the solver can find each of the intermediate values. Just as in a regular LSTM, the hidden state vector of the last timestep represents the output of the LSTM.

Additionally, functions were implemented to handle the scaling of features, as well as capping values between a minimum and maximum value. This capping of values is used in the activation functions to ensure that the resulting values do not exceed the domain, as this would make the problem unsatisfiable. These kind of considerations are important in connecting CP and ML in practice.

Since we want to be able to change the unit size and thus complexity in the LSTM, we need an additional dense layer that maps the LSTM output to the desired dimensionality of our model output, in our specific case the four inside climate targets. The output of our LSTM layer is a vector h_l of LSTM unit size u. The computation of the dense layer is then $\sigma_d(h_l \cdot W_d + b_d)$, where σ_d is the linear capped function, W_d are the weights of the dense layer and b_d the bias.

Put together, the CP model constrains all unknown inside climate variables to be equal to the computed values by the LSTM and dense layer, using the outside weather, actions in the greenhouse, and inside climate of the last l timesteps. A simplified formulation of this constraint is:

$$\underset{i \in I}{X_i(t)} = dense(lstm(\underset{\substack{n \in N \\ t-l \leq t' < t}}{X_n(t')}, W_L, b_L), W_d, b_d) \quad \forall t \in l..T,$$

where X is the complete input array, I the inside climate features, W_L and b_L the trained LSTM weights and biases; N represents all features (inside, actions, and outside), l the chosen time lag, and T the total amount of timesteps taken into account. We used MiniZinc to create the CP model and the LSTM embedding (Nethercote et al. 2007).

Embedding Lintul-3 Model in CP

An adaptation of the Lintul-3 crop model was implemented in MiniZinc using a function that computes a temperature, CO_2, and PAR-light component (see Algorithm 1). An additional input variable LAI is used that represents the Leaf Area Index (an indication of the leaf-size). Some of the coefficients in the function we obtained through trial-and-error using domain knowledge.

Algorithm 1: Simplified MiniZinc code of the adapted Lintul-3 model, where t is temperature, c the CO_2 level, l the PAR-light, and a the Leaf Area Index. Both the function result and included variables are floating numbers.

```
function: lintul(t, c, l, a) =
% temperature
if 6 ≤ t ≤ 14 then (t-6)/8
elseif 14 < t ≤ 28 then 1.0
elseif 28 < t < 40 then (40-t)/12
else 0.0
endif
*
% co2
(1.0 - exp(-0.004 * (1.2 if l ≤ 500 else
    2.0) * c))
*
% par-light
(0.0036 * l) * (1.0 - exp(-0.7 * a)) *
    (1.5 if l ≤ 500 else 0.9) * 12;
```

Economic Model

The profit at the end of the optimisation process – the quantity of interest to the grower – is calculated by subtracting the costs of actions needed to operate a smart greenhouse from the earnings of the crop. This forms the objective function for the CP model. Each action is associated with a cost and the crop growth is associated with a tomato price. Reasonable assumptions of these costs and prices were made using historical data, all normalised to one m^2 greenhouse area. For heating, we computed the amount of gas needed to heat the tubes and linked this to a gas price. For lighting, depending on the intensity and efficiency of the used lighting system, an energy cost could be computed and linked to an energy price. Operating the windows and screens requires so little energy, that these costs were neglected. The used tomato price is based on historical prices and the month of prediction, because it is highly dependent on the time of year in which the produced tomatoes are sold.

Complete CP Model

Below is the complete CP model including constraints that show how the greenhouse and crop models are used together. An LAI of 3.5 was used (this fits the tomato variety); Q represents price; and N the full set of features: outside weather (O), greenhouse actions (A), and inside climate (I).

Table 1 specifies the CP model. The objective is: maximize e_p, and the search guidance to the solver is: $\texttt{integer\_search}(U_A(t))\,\forall t \in [l,T]$. The complete variable input array X_n is:

$$X_n(t) = \begin{cases} K_n(t), & \text{if } n \in O \\ K_n(t) + U_n(t), & \text{otherwise} \end{cases} \qquad \forall n \in N.$$

The JaCoP solver (Kuchcinski and Szymanek 2013) was used to run the CP model because it supports real numbers and has implemented the exponential function, which is used in the activation functions of the machine learning models. The solver performs a sequential search on greenhouse actions, using value randomisation and restarts if no new solutions are found in the search-tree.

Constants	
T	total amount timesteps
k	known amount timesteps
$K_o(t) \quad t = 1,\ldots,T \quad \forall o \in O$	known outside weather
$K_a(t) \quad t = 1,\ldots,k \quad \forall a \in A$	known actions
$K_i(t) \quad t = 1,\ldots,k \quad \forall i \in I$	known inside climate
Decision variables	
$U_a(t) \quad$ t = k+1,...,T $\quad \forall a \in A$	unknown actions
$U_i(t) \quad$ t = k+1,...,T $\quad \forall i \in I$	unknown inside climate
g	crop growth
e_c, e_r, e_p	costs, revenue, profit
Constraints	
$X_i(t) = lstm\ i \in I\ \forall t \in [l,T]$	neural constraint
$g = \sum_{t=l}^{T} lintul(U_I, 3.5)$	crop model
$e_c = \sum_{\substack{\forall a \in A \\ \forall t \in l..T}} U_a(t) \cdot Q_a$	economic costs
$e_r = g \cdot Q_g$	economic revenue
$e_p = e_r - e_c$	economic profit

Table 1: CP model

Lessons Learned

Instantiating functional dependencies As noted, CP is a declarative technology that models dependency between variables through relations. We have found it important to explicitly annotate functional dependencies between variables when possible, i.e., between inputs and outputs of an LSTM. While the MiniZinc compiler is able to derive this functional dependence for small models, in larger models this is not always the case. The benefit of such explicit instantiation is that the solver can simply calculate the value of functionally dependent variables instead of deducing it with expensive techniques, which yields faster solving times.

Functions vs predicates We have found it important to carefully choose between functions and predicates when introducing helper functions, e.g., that encapsulate the crop growth model. MiniZinc supports introduction of helper functions in two ways, namely through predicates and functions. Predicates reify the result of a computation with a new variable through constraints, while functions perform the computation directly and return the result. Predicates therefore introduce additional variables and require computationally more expensive procedures. All LSTM computation within our CP model benefits from using functions over predicates because the computation needs to be performed in a sequential manner – all computation in time step t needs to be executed before the computation in time step $t+1$.

Domain bounds MiniZinc actively tries to bound the variable domains during solving. Tighter bounds typically result in faster solving times (Stuckey, Marriott, and Tack 2020). Often, the bounds of certain variables can be deduced ahead of solving time through domain knowledge. For instance, the hidden state in an LSTM holds values between -1.0 and

1.0, because of the *tanh* activation function. Similarly, the bounds of the cell state can be derived using its equation $c_t = f_t \odot c_{t-1} + i_t \odot c'_t$ and the corresponding activation functions. We incorporate such bounds wherever possible.

Search order The search order over variables is essential in solving CP problems that involves time-based decision variables, as is the case with multi-stage prediction. Since the inside climate at a certain timestep depends on a multitude of variables from previous timesteps, it helps the solver to know which variables to decide first, before moving on to the next. We implemented this using *priority search* (Feydy et al. 2017), letting the solver decide the actions in ascending time order, starting with the first unknown value ($l + 1$).

Empirical Study

We evaluate our system on a real-world use-case in a tomato greenhouse. As the system performance depends on the performance of all its components, we evaluate each component separately. Lastly, we demonstrate the benefit of the DSS by comparing it to decisions made by an experienced grower.

Tomato Greenhouse Dataset

The dataset used to train the greenhouse climate model and initialise the CP model is retrieved from a Venlo-style greenhouse department located in the Netherlands. This department covers an area of 150m^2 and includes LED and SON-T lighting, under- and growtubes (heating), a shading screen, and lee- and windowside ventilation. It contains sensors for each of these actions, as well as the outside weather (radiation, temperature, absolute humidity, wind speed) and the inside climate (PAR-light, temperature, humidity deficit, CO_2). We retrieved a dataset from a tomato variety Merlice cultivation in the 10 month period 25/11/2020 – 20/9/2021. We used the period 28/12/2020 – 4/8/2021 for training, and a summer period 5/8/2021 – 7/8/2021 for prediction.

LSTM Prediction

We validated the LSTM greenhouse model by doing an inside climate prediction in summer. Light, temperature, humidity deficit, and CO_2 are predicted in a multi-stage manner and compared to the real inside climate in the given period. The hyper-parameters that followed from a grid search and were used in this experiment are a batch size of 512, a lag of 6 timesteps (6 hours), and 4 LSTM units. Early stopping was used, which means that the amount of epochs depends on the accuracy on the validation set. The experiment was run 5 times and the result was averaged.

Figure 3 shows that the model is able to predict the temperature and humidity deficit quite well in all five runs, but it has some trouble with predicting the inside PAR light, which can be explained by clouds or other shadows being cast on the PAR-sensors. CO_2, however, is the hardest target feature for the model to predict. Firstly, this department is missing a CO_2 injection sensor. Also, CO_2 levels recorded by the sensor are more fluctuating than that of the other climate variables. Further, how much CO_2 is in the air depends on crop processes, which is not a direct input feature.

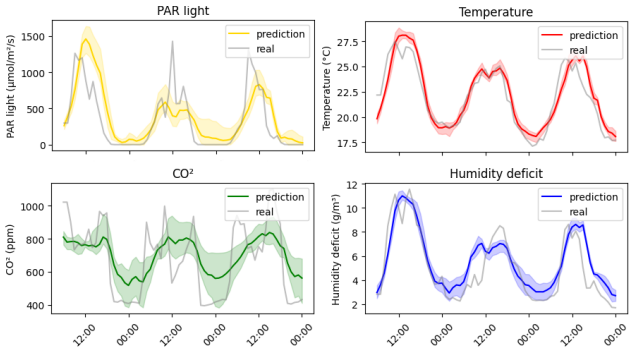

Figure 3: LSTM climate prediction, lag of 6 timesteps, 5/8/2021 00:00 – 7/8/2021 23:00, with an RMSE of 0.0650.

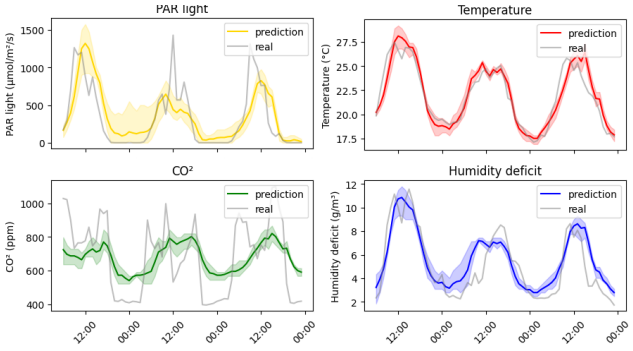

Figure 4: Validation of the greenhouse LSTM model embedded in CP. LSTM-in-CP climate prediction, lag of 6 timesteps, 5/8/2021 00:00 – 7/8/2021 23:00, RMSE 0.0665.

LSTM-in-CP Validation

We now validate the CP model of LSTM. The decision variables of the model, impacting the greenhouse climate, were set manually. This way, the LSTM within the CP model should behave similarly as a regular LSTM. We use the same setup as in the previous experiment.

In Figure 4 we see the LSTM prediction of each of the inside climate variables within CP for a prediction period of 3 days, with an RMSE of 0.0665. The figure shows similar results as the 'regular' LSTM predictions in Figure 3 with a similar RMSE value as well, indicating that the embedded LSTM in CP is behaving as expected.

	Yield (g/m)	Revenue (€/m)	Costs (€/m)	Profit (€/m)
Grower	256.89	0.30	0.01	0.29
DSS	385.31	0.45	0.05	0.40

Table 2: Economic results of the DSS in the period 5/8/2021 00:00 to 5/8/2021 23:00 (one day); timeout 11 hours.

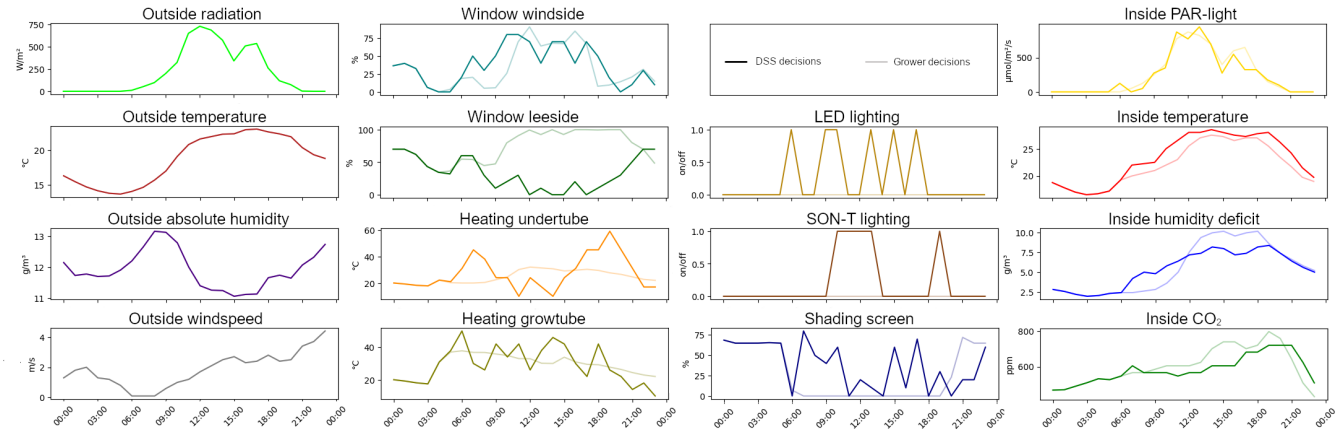

Figure 5: Results of the retrieved outside weather (col 1), greenhouse actions decided by the system (cols 2 & 3), and predicted inside climate (col 4) in the period 5/8/2021 00:00-23:00, timeout 11 hours. These actions and climate are compared to that of the grower. Dark lines represent the result of running the DSS; bright lines represent the grower's decisions and climate.

Comparing Grower and Decision Support System

As the final experiment, we compare the developed system to an experienced grower. We analyse the decisions made by the system and the grower, and the resulting economic outcomes, over the one-day period 5/8/2021. We expect to see the system at least match the grower's profit, and ideally surpass it. Additionally, we also inspect the resources used and the production achieved by both actors, which offers an interesting look into different behaviours.

To obtain the economic outcomes of the grower, we fix the values of the action variables to match the grower's decisions. It is important to note that the climate following the grower's decisions is also a prediction; it is not the actual climate as the LSTM does not have a 100% prediction accuracy. We give our model a timeout of 11 hours to decide on the actions, to give the DSS enough time to surpass the grower's performance. Running the experiment multiple times and averaging the result was not possible in this case, as the resulting intermediate solutions may have very different values for the decision variables due to randomisation.

The resulting decisions and predictions can be seen in Figure 5, the economic results in Table 2. Compared to the grower, the system decides to make more use of the shading screen and lower both the undertube heating and leeside window ventilation, while still reaching a similar temperature; the lighting systems are used more, directly leading to more crop growth. The system thus sees the added benefit of turning the lights on compared to their costs. In the resulting economic values, we can see that indeed the system has higher costs, but has a payback of this in the revenue. The system thus has an increased profit with respect to the grower's decisions. What we can conclude from this experiment, is that given enough time or a small enough search space, the system is able to find better solutions. This shows the potential of such a system. For deployment, however, the runtime is too long and the prediction period too short to be used in practice, motivating our ongoing technical work.

Outlook and Deployment

This paper presents a data-driven optimisation approach for medium-term economic optimisation of greenhouse control. Greenhouse control, and horticulture in general, are an important application area for AI because of the potential for higher productivity and better crop yields with lower use of resources. From a grower's perspective, this would lead to a higher revenue and lower cost of operation. When their profitability is uncertain, for example during an energy crisis, growers could benefit from decision support that takes into account these economic trade-offs. While an initial version of this system is already implemented and connected to live greenhouse data feeds, additional steps are in progress for its full deployment.

The current DSS is able to find a better solution than that of a grower given enough time or with a small enough search space. The next step in addressing the problem complexity is initialising the system with a reasonable solution taken from grower decisions in similar weather conditions, enabling a reduction of the domains. Additional constraints using domain knowledge can reduce the search space even further.

When the solving time is acceptable, further steps include increasing the accuracy of the predictions and performance of the overall system. The simple crop model will be improved either through an LSTM embedding similar to the greenhouse model, or by implementing a more extensive state-of-the-art crop model in CP. When sufficiently accurate, the decision support system can be connected to the greenhouse action actuators, leading towards full autonomous cultivation.

Acknowledgements

Thanks to the IAAI'23 reviewers; S. Andringa, M. Lombardi, and the entire Delphy Digital team. Partially supported by TAILOR (EU GA 952215).

References

Alhnaity, B.; Pearson, S.; Leontidis, G.; and Kollias, S. 2020. Using deep learning to predict plant growth and yield in greenhouse environments. *Acta Horticulturae*, 1296: 425–431.

Ali, A.; and Hassanein, H. 2020. Time-Series Prediction for Sensing in Smart Greenhouses. In *Proc. of GLOBECOM'20*, 1–6.

Bartolini, A.; Lombardi, M.; Milano, M.; and Benini, L. 2011. Neuron Constraints to Model Complex Real-World Problems. In *Proc. of 17th International Conference on Principles and Practice of Constraint Programming (CP'11)*, 115–129.

Bengio, Y.; Lodi, A.; and Prouvost, A. 2021. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2): 405–421.

Challa, H. 1990. Crop growth models for greenhouse climate control. In *Theoretical Production Ecology: Reflections and Prospects*, 125–145.

Chollet, F.; et al. 2015. Keras. https://github.com/fchollet/keras. Accessed: 2022-03-09.

Dai, H.; Khalil, E. B.; Zhang, Y.; Dilkina, B.; and Song, L. 2017. Learning Combinatorial Optimization Algorithms over Graphs. In *Proc. of 31st International Conference on Neural Information Processing Systems*, 6351–6361.

Demirović, E.; Lukina, A.; Hebrard, E.; Chan, J.; Bailey, J.; Leckie, C.; Ramamohanarao, K.; and Stuckey, P. J. 2022. MurTree: Optimal Decision Trees via Dynamic Programming and Search. *Journal of Machine Learning Research*, 23(26): 1–47.

Feydy, T.; Goldwaser, A.; Schutt, A.; Stuckey, P.; and Young, K. 2017. Priority Search with MiniZinc. In *Working Notes of CP'17 Workshop on Constraint Modelling and Reformulation*.

Goyal, K.; Dumancic, S.; and Blockeel, H. 2022. SaDe: Learning Models that Provably Satisfy Domain Constraints. In *Proc. of ECML/PKDD*. Springer.

Hu, H.; Siala, M.; Hebrard, E.; and Huguet, M.-J. 2020. Learning Optimal Decision Trees with MaxSAT and its Integration in AdaBoost. In *Proc. of 29th International Joint Conference on Artificial Intelligence*, 1170–1176.

Iddio, E.; Wang, L.; Thomas, Y.; McMorrow, G.; and Denzer, A. 2020. Energy efficient operation and modeling for greenhouses: A literature review. *Renewable and Sustainable Energy Reviews*, 117: 109480.

Jones, J.; Dayan, E.; Allen, L.; van Keulen, H.; and Challa, H. 1991. A dynamic tomato growth and yield model (TOMGRO). *Transactions of the ASAE*, 34(2): 663–672.

Jung, D.; Kim, H.; Jhin, C.; Kim, H.; and Park, S. 2020. Time-serial analysis of deep neural network models for prediction of climatic conditions inside a greenhouse. *Computers and Electronics in Agriculture*, 173: 105402.

Katzin, D.; van Henten, E.; and van Mourik, S. 2022. Process-based greenhouse climate models: Genealogy, current status, and future directions. *Agricultural Systems*, 198: 103388.

Kuchcinski, K.; and Szymanek, R. 2013. JaCoP – Java Constraint Programming Solver. In *Working Notes of CP'13 Workshop on CP Solvers: Modeling, Applications, Integration, and Standardization*.

Lee, J.; Kang, W.; Moon, T.; Hwang, I.; Kim, D.; and Son, J. 2020. Estimating the leaf area index of bell peppers according to growth stage using ray-tracing simulation and a long short-term memory algorithm. *Horticulture, Environment, and Biotechnology*, 61.

Lombardi, M.; Milano, M.; and Bartolini, A. 2017. Empirical decision model learning. *Artificial Intelligence*, 244: 343–367.

Lopez-Cruz, I.; Fitz-Rodríguez, E.; Torres-Monsivais, J.; Trejo-Zúñiga, E.; Ruiz Garcia, A.; and Arias, A. 2013. Control Strategies of Greenhouse Climate for Vegetables Production. *Biosystems Engineering: Biofactories for Food Production in the Century XXI*, 401–421.

Morrow, R. 2008. LED lighting in horticulture. *HortScience*, 43(7): 1947–1950.

Nemali, K. 2022. History of Controlled Environment Horticulture: Greenhouses. *HortScience*, 57(2): 239 – 246.

Nethercote, N.; Stuckey, P. J.; Becket, R.; Brand, S.; Duck, G. J.; and Tack, G. 2007. MiniZinc: Towards a Standard CP Modelling Language. In *Proc. of 13th International Conference on the Principles and Practice of Constraint Programming (CP'07)*, 529–543.

Rodriguez, F.; Berenguel, M.; Guzmán, J.; and Arias, A. 2015. The Greenhouse Dynamical System. *Advances in Industrial Control*, 9–97.

Shibu, M.; Leffelaar, P.; van Keulen, H.; and Aggarwal, P. 2010. LINTUL3. *European Journal of Agronomy*, 32(4): 255–271.

Stanghellini, C. 1987. *Transpiration of greenhouse crops: an aid to climate management*. PhD thesis, Wageningen University.

Stuckey, P. J.; Marriott, K.; and Tack, G. 2020. The MiniZinc Handbook. https://www.minizinc.org/doc-2.5.5/en/index.html. Accessed: 2022-03-09.

van Straten, G.; Challa, H.; and Buwalda, F. 2000. Towards user accepted optimal control of greenhouse climate. *Computers and Electronics in Agriculture*, 26(3): 221–238.

van Straten, G.; and van Henten, E. 2010. Optimal Greenhouse Cultivation Control: Survey and Perspectives. *IFAC Proceedings Volumes*, 43(26): 18–33.

van Weel, P.; Geelen, P.; and Voogt, J. 2018. *Plant Empowerment: The Basic Principles*. Plant Empowerment Academy.

Vanthoor, B. 2011. *A model-based greenhouse design method*. Ph.D. thesis, Wageningen University.

Verhaeghe, H.; Nijssen, S.; Pesant, G.; Quimper, C.-G.; and Schaus, P. 2020. Learning Optimal Decision Trees using Constraint Programming. In *Proc. of 29th International Joint Conference on Artificial Intelligence*, 4765–4769.

Wallace, M. 1996. Practical applications of constraint programming. *Constraints*, 1(1–2): 139–168.